

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188	
<small>Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate to any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.</small>				
1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE 30 October 2001		3. REPORT TYPE AND DATES COVERED Final Report 26 April 2001 to 31 October 2001
4. TITLE AND SUBTITLE Compressed Internet Protocol (IP) Data via Geosynchronous Earth Orbit (GEO) Satellite Circuits			5. FUNDING NUMBERS (C) Contract Number N00039-01-C-3208  (PR) Project Number N00039-01-PR-D61106	
6. AUTHOR(S) Mr. Stuart W. Card, Mr. Eric R. Crossman, Ms. Christine M. Haka, Mr. Youngki Hwang, Mr. Michael W. Joseph, Mr. George H. Palmer, Mr. Fred W. Tims				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Critical Technologies Inc. Suite 400, Technology Center 4th Floor, 1001 Broad Street Utica, NY 13501			8. PERFORMING ORGANIZATION REPORT NUMBER  ATC-P1-FR	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Space and Naval Warfare Systems Command 4301 Pacific Highway San Diego, CA 92110-3127			10. SPONSORING/MONITORING AGENCY REPORT NUMBER	
11. SUPPLEMENTARY NOTES Attached report version 1.2 supersedes all previously submitted drafts and is the final report for the Phase I base effort; if the Phase I option is exercised, this report will be revised accordingly and resubmitted at the conclusion of the option effort.				
12a. DISTRIBUTION/AVAILABILITY STATEMENT Proposed availability statement, subject to Government approval -- Approved for public release; distribution unlimited.			12b. DISTRIBUTION CODE	
13. ABSTRACT (Maximum 200 words) To enable Internet Protocol (IP) applications in the submarine environment, Critical Technologies Inc. (CTI) has defined requirements for an integrated system, comprising enhanced Data Link, Network and Transport protocols (OSI Layers 2-4), and an innovative Application Traffic Controller (ATC, operating in OSI Layers 5-7). The system will optimize efficiency and effectiveness of fleet communications, despite submarine SATCOM's high Bit Error Rate, low bandwidth, long latency, extreme channel variability and severely intermittent connectivity. The ATC integrates caching, compression, prioritization and queuing, under control of system-administrator specified rules, to optimize link utilization. The system is also designed to operate under Low Probability of Detection / Intercept and EMCON (with planned and unplanned blackouts, in various constrained 2- and 1-way communications postures), by integrating enhanced protocols that provide reliable transport of IP traffic to submarines without acknowledgement. The system will benefit various assets of the current and future US fleet, as well as other military and commercial organizations. These assets could include fast attack and ballistic missile submarines, surface ships, airborne platforms, unmanned vehicles, littoral elements, and various land, sea, air and space networks.				
14. SUBJECT TERMS submarine communications, SATCOM, EMCON, LPD, LPI, Data Link, Internet, Transport, protocol, TCP/IP, caching, compression, prioritization, queuing, QoS			15. NUMBER OF PAGES 66 inc. title, plus this SF298 16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED	18. SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED	19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	20. LIMITATION OF ABSTRACT UL	

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)  
Prescribed by ANSI Std. Z39-18

20011105 021

Navy Small Business Innovation Research (SBIR) Program

Compressed Internet Protocol Data Via Geosynchronous Earth Orbit Satellite Circuits

**N01-052 Phase I Technology Report Study (Final Report)**

CDRL Item Number: A002

2001-Oct-30

Sponsored by:

Space and Naval Warfare Systems Command

Attn: Kurt Reese PMW 173

4301 Pacific Highway

San Diego, CA 92110-3127

Contract Number: N00039-01-C-3208

Contract Dollar Amount \$69,856

Competitive Award

Prepared by:

Stu.Card@critical.com, Eric.Crossman@critical.com, Christine.Haka@critical.com, Youngki.Hwang@critical.com,

Mike.Joseph@critical.com, George.Palmer@critical.com, Fred.Tims@critical.com

Critical Technologies Inc.

Suite 400 Technology Center

4<sup>th</sup> Floor 1001 Broad Street

Utica, NY 13501

315-793-0248 / FAX -9710

<http://www.critical.com>

Proposed availability statement, subject to Government approval:

Approved for public release; distribution unlimited.

Security Markings or Handling Restrictions: N/A

1	Scope & Background including Physical (OSI Layer 1).....	4
1.1	Document Scope and Project Background.....	4
1.2	Environmental Characteristics.....	4
1.3	Intermittent and EMCON Operations.....	5
2	Data Link (OSI Layer 2).....	6
2.1	Section Scope and Background. ....	6
2.2	General Requirements. ....	6
2.3	Detailed Capability Requirements. ....	6
2.3.1	<i>Connection Management</i> .....	6
2.3.1.1	Dial-on-Demand / Auto-Answer.....	7
2.3.1.2	Link Admission Control.....	7
2.3.1.3	Addressing. ....	7
2.3.2	<i>Wireless Medium Access Control (MAC)</i> .....	7
2.3.2.1	Requirements extracted from "Submarine Requirements for ADNS".....	7
2.3.2.2	Requirements extracted from "Revision of MIL-STD-188-184...".....	7
2.3.3	<i>Logical Link Control (LLC)</i> . ....	8
2.3.3.1	Length Indication.....	8
2.3.3.2	Flow Control. ....	8
2.3.3.3	Errored LLC-PDU Discard or Delivery w/Error Indication. ....	8
2.3.3.4	LLC-PDU Residual Error Detection.....	8
2.3.4	<i>Segmentation And Reassembly (SAR)</i> . ....	8
2.3.4.1	[In-order delivery]. ....	8
2.3.4.2	Segmentation. ....	8
2.3.4.3	Reassembly. ....	8
2.3.5	<i>Automatic Retransmission reQuest (ARQ)</i> .....	8
2.3.6	<i>FEC (see also general discussion of coding 3.3)</i> . ....	9
2.3.6.1	Errored FEC-PDU Discard or Delivery w/Error Indication. ....	9
2.3.6.2	FEC-PDU Residual Error Detection. ....	9
2.3.6.3	Forward Error Correction (FEC) [and interleaving]. ....	9
2.3.7	<i>Framing</i> .....	9
2.3.7.1	[Scrambling]. ....	9
2.3.7.2	Bit Stuffing or Data Link Escape (DLE). ....	9
2.3.7.3	Preamble/Postamble Insertion/Extraction. ....	9
2.3.7.4	PHY XMT/RCV Timing.....	9
2.4	Unacknowledged Operation for EMCON.....	10
2.5	Compression (see also general discussion of coding 3.3). ....	10
2.6	Research to Exploit Characteristics of this Special Case.....	11
2.7	Referenced Documents.....	11
2.7.1	<i>Government Documents</i> . ....	11
2.7.2	<i>IETF Proposed Standards (<a href="http://www.rfc-editor.org">http://www.rfc-editor.org</a>)</i> . ....	11
3	Network (OSI Layer 3).....	12
3.1	Section Scope and Background. ....	12
3.2	Classification. ....	12
3.3	Coding.....	12
3.4	Packet Queuing.....	13
3.5	Concurrent Multipath Routing.....	13

4	Transport (OSI Layer 4) .....	14
4.1	Section Scope and Background .....	14
4.2	What TCP Is .....	14
4.3	Why TCP Is Used.....	15
4.3.1	<i>To Communicate, a Common Language is Needed.....</i>	15
4.3.2	<i>How Good is Good Enough? .....</i>	15
4.3.2.1	Currency.....	15
4.3.2.2	Integrity. ....	15
4.4	Roughly How TCP Works.....	15
4.5	TCP Problems.....	16
4.6	Dealing With The Problems.....	16
4.6.1	<i>Addressing The Low Bandwidth Problem - Multiple Paths (Layer 3). ....</i>	16
4.6.2	<i>Addressing The TCP Confusion Problem – Data Link Automatic Repeat reQuest.....</i>	17
4.6.2.1	It Disguises Loss as Latency.....	17
4.6.2.2	It Increases the Incidence of OOOD (Out-Of-Order Delivery). ....	17
4.7	Dealing With The Side Effects of The Solutions.....	18
4.7.1	<i>The Side Effects .....</i>	18
4.7.1.1	The Increased Out-Of-Order Problem.....	18
4.7.1.2	The Additional Opportunities for Confusing TCP.....	19
4.7.2	<i>Solution Approaches .....</i>	19
4.7.2.1	Tune Standard TCP in the Endpoints.....	19
4.7.2.2	Create a Gateway Using Standard TCP. ....	19
4.7.2.3	Use Something Other Than Standard TCP in The Endpoints. ....	19
4.7.2.4	Create a Gateway Using Something Other Than Standard TCP. ....	20
4.8	Additional Requirements .....	21
4.8.1	<i>Multicast.....</i>	21
4.8.2	<i>EMCON.....</i>	21
4.8.2.1	Multiple Transmissions.....	21
4.8.2.2	Forward Error Correction (FEC). ....	22
4.9	Looking For Solutions.....	22
4.9.1	<i>TCP Candidates .....</i>	22
4.9.2	<i>Reliable Multicast Candidates. ....</i>	23
4.10	Recommendations .....	24
4.11	Background Information Concerning TCP.....	25
4.11.1	<i>TCP Characteristics Table.....</i>	25
4.11.2	<i>TCP Reference Table. ....</i>	26
4.12	References.....	26



5	Session / Presentation / Application (OSI Layers 5-7) .....	29
5.1	Section Scope and Background .....	29
5.2	Compression .....	29
5.3	Caching .....	30
5.3.1	<i>Other Considerations</i> .....	32
5.3.1.1	Minimizing Burden on Wireless Link .....	32
5.3.1.2	Complications in Data/Information Acceptability Decisions .....	32
5.4	Prioritization and Queuing .....	34
5.4.1	<i>Message Classifications And Hierarchy</i> .....	34
5.4.2	<i>Prioritization within the Classifications (forwarding priority)</i> .....	36
5.4.3	<i>Queue Size (drop priority)</i> .....	36
5.5	Implementation of Prioritization and Queuing .....	37
5.5.1	<i>Linux Netfilter</i> .....	37
5.5.2	<i>Linux traffic controller (tc)</i> .....	37
5.5.3	<i>Implementation of Priority Scheme</i> .....	37
5.6	User interface .....	38
5.7	Supplemental Support .....	47
5.8	Prototyping tools .....	47
5.8.1	<i>Background</i> .....	47
5.8.1.1	QoS Under Linux .....	47
5.8.1.2	Proxy Servers under Linux .....	48
5.8.2	<i>Experimentation in Feasibility</i> .....	48
5.8.2.1	Testbed Configuration .....	48
5.8.2.2	Web Caching .....	48
5.8.2.3	Traffic Control .....	49
5.9	References .....	49
6	Summary of Preliminary Conclusions .....	50
	Appendix A: Glossary of Acronyms and Terms .....	51
	Appendix B: Proxy Caching Verification Tests .....	54
	Appendix C: APACHE WEB SERVER with proxy+cache+offline HOWTO .....	59
	Appendix D: Linux IP Filtering – Comparison of IP Chains and IP Tables .....	64

# 1 Scope & Background including Physical (OSI Layer 1).

## 1.1 Document Scope and Project Background.

This report applies to the Compressed Internet Protocol Data Via Geosynchronous Earth Orbit Satellite Circuits project. The purpose of this project is to investigate and define requirements for an integrated system, comprising enhanced Data Link and Transport layers, and an innovative Application Traffic Controller (ATC), that enables IP networking despite poor environmental characteristics and severe operational conditions. The general nature of the system is a wireless IP network gateway. The project is currently in SBIR Phase I. The primary sponsor is SPAWAR. The primary intended users are fast attack submariners. The developer is Critical Technologies Inc. (CTI). Support agencies are TBD. The initial test site is CTI's laboratory. Other relevant documents are listed as references at the ends of the sections. The purpose of this document is to summarize preliminary conclusions, both analytic and empirical. There are no security or privacy considerations associated with the use of this document.

## 1.2 Environmental Characteristics.

The requirement for stealthy global operations limits available communications channels, presently to bent-pipe relays aboard geostationary military satellites, mostly operating in the Ultra High Frequency (UHF) band with relatively narrow channels (UHF MILSATCOM). Use of small antennas, extended just above the waves, from a rolling platform, degrades signal quality. The Physical (OSI Layer 1) channel thus has high and unstable error rate (Bit Error Rate [BER] typically from  $10^{-6}$  to  $10^{-5}$  and worse), low bandwidth (16000 Bits Per Second [bps] shared or 2400 bps dedicated), and long latency (250 milliseconds speed of light lag inherent in a single geosynchronous satellite hop, plus switching and cryptographic delays, doubled for the round-trip). These characteristics are worse than those assumed by the Internet Protocol suite designers, and much worse than those assumed by the developers of the Commercial Off The Shelf (COTS) / Government Off The Shelf (GOTS) IP application software used both ashore and shipboard (including that comprising most of the GOTS Delta Load). These environmental characteristics define the basic system requirements.

The system should provide multiple access Medium Access Control (MAC) functionality to exploit half-duplex channels. It should also exploit full-duplex channels, and provide at least a limited capability over simplex channels (RFC 3077). It should exploit very low data rate channels and minimize bandwidth overhead. It should tolerate very long channel latencies (approximately 1 second for link encrypted narrowband UHF MILSATCOM), and should minimize whatever additional latencies it introduces. The system should tolerate, and (from the perspective of the applications and users) significantly ameliorate, very high error rate channels (random BER of  $10^{-6}$  to  $10^{-4}$  plus burst errors, dropouts, etc.): this implies use of adaptive strength Forward Error Correction (FEC); and when EMCON does not preclude acknowledgments, assured delivery based upon Automatic Retransmission reQuest (ARQ).

The system should tolerate highly asymmetric channel characteristics: this implies that its parameters (ARQ enabled/disabled, FEC code strength, maximum SDU length, maximum PDU length, maximum PDU retransmissions, maximum channel interruption without broken link, etc.) should be tunable independently in each direction. It should tolerate dynamically varying channels. Its parameters should automatically adapt during the course of a session without the need to break and re-establish connections.

The system should operate over UHF MILSATCOM Demand Assigned Multiple Access (DAMA) and Demand Assigned Single Access (DASA) channels, and exploit rather than duplicate their addressing and other functions. It should provide auto-answer and dial-on-demand capabilities coupled with their order-wire operations, and not break link or lose data due to interruptions from normal operations (order-wire changes, etc.). It should flow Quality of Service (QoS) indicators up and down the stack, from the Application through the Physical layers, so that lower layer information can be used in higher layer processing (such as queue management) and higher layer information can be used in lower layer processing (such as DAMA resource allocation by the Network Control Terminal).

### ***1.3 Intermittent and EMCON Operations.***

Severe operational conditions are imposed by doctrine: communication is limited to a few minutes every few hours. During most of these brief sessions, the submarine is under Low Probability of Detection (LPD) / Low Probability of Intercept (LPI) or other Emissions Control (EMCON) restrictions, and thus can neither carry on a two-way conversation, nor even acknowledge receipt of the shore station's transmissions. Given the large coverage footprints of geostationary SATCOM relative to the distance that a submarine could travel during even a long surface interval, no need for automatic handoffs is foreseen. Nonetheless, intermittent connectivity is the major issue in this effort. The system should detect lost link rapidly and reliably, indicate the loss of link to applications and administrators, re-establish link rapidly and automatically when possible, indicate when link has been [re]established, and provide QoS information regarding the established link. The system should provide estimates of time remaining to the next predictable state change (for example: link going down due to rain fade, inferred from the trend in measured error rate; or link expected to come up, based upon scheduled rise to periscope depth to receive broadcasts of interest per the ZBO). Unlike the other features listed herein, each of which is found in at least some existing protocol stacks, this proposed feature is unique.

## 2 Data Link (OSI Layer 2).

### 2.1 Section Scope and Background.

Although the Data Link is not the focus of this project, it is a critical element of the system, and interacts significantly with all other layers. Use of a suitable Data Link is assumed in ATC analysis and design; its assumed characteristics follow (extrapolated from requirements defined under the USAF Information For Global Reach program).

### 2.2 General Requirements.

To efficiently provide bandwidth-on-demand for the dynamic and asymmetric offered traffic load, the Data Link should support dynamic (near real-time) [re]allocation of link resources, independently in each direction (up- and down-link). Because offered load will generally exceed available link bandwidth, it should allocate resources and pre-empt previous allocations per system administrator configured prioritization policies. To minimize TCP 'misbehavior', it should provide assured delivery. To provide expedited delivery of the time-critical traffic typically carried by UDP (versus the quality-critical traffic typically carried by TCP), to minimize latency variation (at the expense of increased packet loss), and to enable data transport during EMCON operation, it should also provide a non-assured (unacknowledged) mode of operation. To ensure reliability, especially when operating without acknowledgements, it should provide adaptive FEC. To efficiently support multipoint distribution, it should provide true multi- and broadcast. It should support an Address Resolution Protocol [ARP]. It should support QoS or at least relative priorities.

There must be mechanisms for detecting: when the local Data Link state machine has entered a 'silly state' (any state not in the set of intentionally visited states, but nonetheless potentially reachable, generally through error); when the remote Data Link state machine has entered a silly state; and when the local and remote Data Link state machines have lost mutual synchronization. There must be mechanisms: for resetting the local Data Link state machine to a 'clear' initial state, from which it can recover synchronization with the remote Data Link state machine; and for indicating to the remote Data Link state machine that it should reset itself to a clear initial state. There should be mechanisms for recovering synchronization between the local and remote Data Link state machines, whenever possible, without requiring such a reset. There must be mechanisms: for flushing the data buffers of the local Data Link protocol entity (for example, to clear them of traffic which has aged excessively while the link was down); for requesting that the remote Data Link protocol entity flush its data buffers; for indicating, to the higher layers of the local system, that a buffer has been flushed; and, to the extent feasible, for obtaining operator confirmation that an indicated buffer flush should be performed. There may be a mechanism for indicating that a Data Link PDU carries 'urgent data' which should be processed immediately by the remote Data Link protocol entity, rather than consecutively in the sequence in which it was received with non-urgent data.

### 2.3 Detailed Capability Requirements.

From the overall Data Link Layer requirements identified above, detailed capability requirements can be derived and allocated to sublayers. The derivation and allocation presented herein is not unique, but should be generally representative of suitable Data Links, at least with respect to the list of functions required, if not to their partitioning. Each function listed in this section is found in at least some existing Link Layer protocols, although not all are found together. This section addresses two-way operation as the baseline; the next section (2.4) addresses EMCON.

#### 2.3.1 Connection Management.

As the Network (Layer 3) protocol to be supported, IP, is connectionless, but the requirement for a reliable Link Layer implies that the Data Link (Layer 2) protocol will be connection-oriented, the Data Link must provide transparent (to the upper layers) connection management. Note: the use of the phrase 'connection management' is consistent with industry jargon, despite its being not a Management Plane but rather a Control Plane function.

### 2.3.1.1 Dial-on-Demand / Auto-Answer.

A capability like that of a Point-to-Point Protocol (PPP) daemon configured for 'on demand' operation is required: when in the disconnected state, upon receipt of a request to prepare to pass traffic, the Data Link must initiate a connection. Such a request might be triggered by the appearance of traffic in a queue, by an automated control process following a transmission schedule, by a console command from a human operator, by a Management Plane entity, etc. The Data Link should also support opportunistic connection establishment based upon indications from the Physical Layer of a newly available channel. Upon receipt of an indication or confirmation of a new connection or disconnection, the Data Link should notify the Network (Layer 3) to facilitate updates to the routing tables.

### 2.3.1.2 Link Admission Control.

The Data Link should provide Link Admission Control, refusing traffic flows that have not received Wireless MAC resource allocations (2.3.2). It may also provide policing, dropping (per specified drop policies) some of the packets of flows which attempt to exceed their resource allocations; but shaping and policing functions generally are assumed to be provided by the Network (Layer 3) using Differentiated Services (RFC 2475) or the like.

### 2.3.1.3 Addressing.

Source and destination addresses unique within link scope must be affixed to CM-PDUs. To ensure uniqueness, they must either be globally unique or dynamically bound. The Data Link also should provide the capability of establishing multiple logical links on a single physical medium, each carrying a single flow, each using its own Wireless MAC sublayer resource allocation (2.3.2). If the capability of establishing multiple logical links is provided, the Data Link must provide addressing adequate to distinguish them. This may be provided solely through source and destination station addressing. Provided this can be achieved without excessive overhead, it would be preferable to support multiple logical links between the same source and destination stations, as this enables QoS, which is required to balance a need for explicit traffic prioritization and corresponding bandwidth allocation (2.3.2) with a need to avoid starvation of low-priority flows (leading to application timeout, data loss and wasted bandwidth).

## 2.3.2 Wireless Medium Access Control (MAC).

The Data Link must support bandwidth management and prioritized traffic handling by providing QoS-aware (or at least prioritized) Channel Admission Control and multiple access Medium Access Control for the wireless medium. This sublayer must satisfy not only the generic requirements for a wireless MAC protocol, which are substantial and will not be detailed here (until Phase II), but also the following user-unique documented requirements...

### 2.3.2.1 Requirements extracted from "Submarine Requirements for ADNS".

The MAC protocol must support simultaneous access for a minimum of 4 submarines. If only a single submarine is attempting to use a particular SATCOM channel, it must establish network connectivity within 30 seconds of establishing RF connectivity; if 4 submarines are attempting simultaneously to use the same channel, network connectivity must be established for each within 2 minutes of establishing RF connectivity. MAC latencies affect the latencies of establishing both RF and network connectivity, so their allocation may be subtle. Submarine access to the medium must regard priorities based upon various mission need factors tied to submarine identification.

### 2.3.2.2 Requirements extracted from "Revision of MIL-STD-188-184..."

The MAC protocol must not depend upon one mobile node being able to hear other mobile nodes, only the controlling station; thus it must use a non-token based Time Division Multiple Access design. A minimum of 8 simultaneously active mobile nodes must be supported (versus the 4 required immediately above). The highest priority current user must be able to 'capture' the entire channel for exclusive use. The number and length of time slots must be automatically and dynamically adapted based upon number of active nodes, priorities, etc. The mobile nodes must not be restricted to using the 'lowest common denominator' modulation, coding and data rate. Interrupted traffic flows must resume without loss upon return from priority interruptions (*this requirement, to be meaningful, must apply to all layers of the stack, but will not be repeated in each section*).

### 2.3.3 Logical Link Control (LLC).

#### 2.3.3.1 Length Indication.

The LLC must prepend a length field (and may insert Ethertype or other fields) to the LLC-SDU as part of forming the LLC-PDU, and must make the received length information available to the higher layers (RFC 1812).

#### 2.3.3.2 Flow Control.

As the link bandwidth will typically be exceeded by traffic demand, will vary dynamically with channel characteristics and will fluctuate due to Link Layer retransmission, the Data Link should provide single-hop flow control. If the capability of establishing multiple logical links is provided, flow control should operate on each logical link (2.3.1.3).

#### 2.3.3.3 Errored LLC-PDU Discard or Delivery w/Error Indication.

There should be a mode control flag to select (on a logical link specific basis if the capability of establishing multiple logical links is provided) whether LLC-PDUs (reassembled SAR-SDUs) that have been detected as in error will be discarded or delivered as LLC-SDUs to the higher layers.

#### 2.3.3.4 LLC-PDU Residual Error Detection.

The Data Link must provide means of detecting residual errors in LLC-PDUs (reassembled SAR-SDUs). This should reduce the frequency of undetected errored LLC-PDUs to at most one (1) per day per link operating continuously at maximum rate. A typical means would be a 32-bit Cyclic Redundancy Check.

### 2.3.4 Segmentation And Reassembly (SAR).

#### 2.3.4.1 [In-order delivery].

The SAR sublayer may provide in-order delivery of its SDUs. It need not provide this function, as the sole Network (Layer 3) protocol for which support is required is IP, which does not require in-order delivery. If the sublayer does provide in-order delivery, it must provide: mechanisms (presumably via the Management interface) for disabling the function and for determining whether the function is currently enabled; and a documented default state of the function (enabled or disabled).

#### 2.3.4.2 Segmentation.

The Data Link must segment SAR-SDUs into SAR-PDUs small enough to be protected by efficient block FEC and efficiently retransmitted when uncorrectable or entirely lost. Segment size should be variable, indeed should be automatically adapted (based upon recent FEC and ARQ history).

#### 2.3.4.3 Reassembly.

The Data Link must reassemble SAR-PDUs into SAR-SDUs.

### 2.3.5 Automatic Retransmission reQuest (ARQ).

ARQ should be performed on a SAR-SDU specific basis, so that SDUs that have been completely received can be delivered, without waiting for retransmission of SAR-PDUs not part of those SDUs; this requirement is weakened if in-order delivery is provided. Selective [Negative] ACKnowledgement should be used. ARQ must retransmit not only errored PDUs (detected by 2.3.6.2, when the PDU can be identified with high confidence) but also missing PDUs; these must be detected.



## 2.3.6 FEC (see also general discussion of coding 3.3).

### 2.3.6.1 Errored FEC-PDU Discard or Delivery w/Error Indication.

There should be a mode control flag to select (on a logical link specific basis if the capability of establishing multiple logical links is provided) whether FEC-PDUs that have been detected as in error will be discarded or delivered to the ARQ sublayer.

### 2.3.6.2 FEC-PDU Residual Error Detection.

The Data Link must provide means of detecting residual errors in FEC-PDUs. These means should reduce the frequency of undetected errored FEC-PDUs to at most one (1) per day per link operating continuously at maximum rate. A typical means would be a 32-bit Cyclic Redundancy Check. Alternatively the means may be a capability of the FEC function itself to detect uncorrectable blocks.

### 2.3.6.3 Forward Error Correction (FEC) [and interleaving].

The Data Link must provide strong block FEC of FEC-PDUs. This should reduce the frequency of uncorrectable ARQ-PDUs to at most one (1) per maximum size SAR-SDU. A typical means would be a Reed-Solomon code with the capability for automatically increasing and decreasing the number of check bytes in response to corrections reported by the far end. Turbo codes should also be considered, but varying their strength is less straightforward.

## 2.3.7 Framing.

### 2.3.7.1 [Scrambling].

The Framing sublayer may provide scrambling to minimize the likelihood of long repetitive symbol sequences (which might otherwise cause loss of synchronization at some level). It need not provide this function, as Data Communications Equipment (DCE) that requires scrambling, should provide it. Furthermore, the higher layers are assumed to provide compression and/or encryption, either or both of which will ensure high entropy in the data stream. If the sublayer does provide scrambling, it must provide: mechanisms (presumably via the Management interface) for disabling the function and for determining whether the function is currently enabled; and a documented default state of the function (enabled or disabled).

### 2.3.7.2 Bit Stuffing or Data Link Escape (DLE).

Detection of the start of a frame typically involves the use, for framing purposes only, of a binary sequence (synchronous) or a character (asynchronous) that is not allowed to appear elsewhere. Since higher layers may demand transmission of arbitrary data sequences, there must be a mechanism for bit stuffing (synchronous) or Data Link Escape (DLE, asynchronous) to prevent user data being erroneously decoded as framing.

### 2.3.7.3 Preamble/Postamble Insertion/Extraction.

There must be a means for marking (in the sender) and detecting (in the receiver) the start of a transmission frame. This must NOT be based solely on nominal transmission slot timing. There must be a means for marking (in the sender) and detecting (in the receiver) the end of a transmission frame. This may be based solely on frame length information encoded in the frame.

### 2.3.7.4 PHY XMT/RCV Timing.

The transmission slot timing allocations determined by the Wireless MAC entity (2.3.2) must be communicated to the lowest layers of the stack, to enable accurate positioning of the actual, relative to the nominal, start of transmission. Received frames must be time stamped as accurately and precisely as the software environment permits.

## 2.4 *Unacknowledged Operation for EMCON.*

One of the two major foci of this project is identification of techniques for *reliable* versus traditional *assured* delivery. Assured delivery uses acknowledgements to confirm receipt, but transmission of ACKs would violate radio silence. Reliable delivery uses FEC, interleaving, etc. to achieve high confidence of, but cannot confirm, receipt. Reliable techniques must be used at both the Link and Transport layers to enable system-level reliable EMCON operation.

In the absence of ACKs, reliability can only be provided by redundancy (bandwidth expansion). This can be: simple repetitive transmission of uncoded data; repetitive transmission of data encoded for error detection; single transmission of data encoded for error correction; or, ultimately, repetitive transmission of data encoded both for error correction and for detection of uncorrectable errors. In the context of repetitive transmission, the distinction between error detection and error correction blurs.

Burst error tolerance can be improved by interleaving, especially in the case of repetitive transmission.

The Multicast Dissemination Protocol (MDP) uses these techniques at or above the Transport layer, as do some other reliable multicast and reliable UDP protocols. Although they were developed for multicast, they may in some cases (including, it would appear, this one) be appropriate for unicast also. If these techniques are not already used also at the Link Layer, their adoption should be considered on a 'fast track'. In Phase II, based upon analysis of the Government's current Data Link, if these techniques are not already exploited in the Link Layer, an approach for their integration will be devised and its cost/benefit estimated.

## 2.5 *Compression (see also general discussion of coding 3.3).*

Compression performed at the Physical or Data Link Layer operates on nearly all of the transmitted data. This includes the nested headers of all the layers above that at which compression is performed, maximizing the benefit of the compression performed there, but allowing use only of low compression ratio (lossless) techniques. Benefit is also limited to compressible data: that which has not been previously compressed or encrypted. Use of DCE hardware based lossless compression is recommended, but only when channel quality is sufficient that the effects discussed next manifest themselves at tolerable levels.

Compression interacts with error control. Decompression of uncorrected errored data can lead at best to expansion of the errors, and at worst to indefinitely prolonged propagation of the errors in the decompressed data stream. The lower in the stack compression is performed, the worse is this effect, as it corrupts not only application data but also protocol headers. Compression in the Link Layer is thus not recommended, unless all of the following conditions prevail: Application Layer compression is infeasible or inadequate, or data packets are so short that protocol header overhead dominates; data has not been previously encrypted or otherwise rendered incompressible; DCE hardware based lossless compression is absent; and channel quality is adequate to avoid frequent decompression-induced propagation of errors, especially into protocol headers (RFCs 1144, 2507 and 2508).

If this case is common, in Phase II, equipped with the Government's choice of a Data Link, the crossover point where channel quality is adequate to support Link Layer compression will be identified, through quantitative analysis, simulation and/or experimentation.



## 2.6 Research to Exploit Characteristics of this Special Case.

The case of a submarine coming infrequently to periscope depth for brief periods stretches the meaning of the phrase 'intermittent operation': elsewhere, this indicates general connectivity, with occasional brief disconnections; here, it means general non-connectivity, with occasional connections. While this poses significant problems for standard network protocols, it also offers a unique opportunity.

Much effort in the design of modulation and coding goes into devising schemes that can be rapidly and efficiently decoded using 'greedy' algorithms that are hoped to provide *nearly* optimal performance most of the time; simpler algorithms that can be guaranteed to provide optimal performance tend to require exponential time and/or memory, which is typically unacceptable. Here, however, given several hours submerged, during which the signals received during the most recent brief surface interval can be intensively 'studied', such exponential algorithms may be feasible. Issues involving decryption and multiple security enclaves may conspire to limit this opportunity: information from Layer 3 and above will be unavailable to assist in this process; but information from Layer 2 and below, confined to the radio room, should be sufficient to enable significant gains, if such gains are feasible at all.

Signals can be recorded as raw data, and then subjected to massive batch processing. Iterative algorithms can exploit 'soft decision' information and gradually adjust decision thresholds (successively trying different assumptions regarding environmental noise, etc.). Adaptive filters can be afforded extremely long convergence times. The repetitive (or otherwise highly redundantly encoded) transmission required for EMCON synergistically enhances this opportunity. While the receiver does not know in advance what signals it will receive, it can with high confidence identify repetitive received sequences. The knowledge that several different but similar received sequences are differently corrupted repetitions of a single sequence is very powerful. It affords not only error correction but also channel characterization. Subsequent processing can use a channel model with the freshly acquired parameters for equalization, more nearly optimal soft decision decoding, etc. Clearly, this is not useful for information that is required in [near] real-time; however, it is expected that at least some messages are usable if decoded anytime during the several hours at speed and depth, between one periscope depth SATCOM connection and the next.

Again, the Link Layer is not the focus of the current effort; but the gains potentially realizable through research in this area are exciting. These topics could be investigated through expansion of project scope in Phase II or elsewhere.

## 2.7 Referenced Documents.

### 2.7.1 Government Documents.

Submarine Requirements For ADNS, 2001-Mar-22, prepared for SPAWAR PMW 158-1.

Revision of MIL-STD-188-184 to include a Dynamic TDMA Mode and an I/P (*sic*) Interface Protocol, Charles Gooding and David Aitken; SPAWAR Systems Center and Strategic Partnerships International

### 2.7.2 IETF Proposed Standards (<http://www.rfc-editor.org>).

RFC 1144	Compressing TCP/IP headers for low-speed serial links	1990-Feb
RFC 1812	Requirements for IP Version 4 Routers	1995-Jun
RFC 2475	An Architecture for Differentiated Services	1998-Dec
RFC 2507	IP Header Compression	1999-Feb
RFC 2508	Compressing IP/UDP/RTP Headers for Low-Speed Serial Links	1999-Feb
RFC 3077	A Link-Layer Tunneling Mechanism for Unidirectional Links	2001-Mar

### 3 Network (OSI Layer 3).

#### 3.1 Section Scope and Background.

Work at Layer 3 was not contemplated in the Phase I proposal. However, in empirical validation of the tools and techniques proposed for use in the prototype, it became apparent that *de facto* standard mechanisms for classifying traffic (required to support ATC functions in other layers) operate in this layer. Also, in study of the network topology diagrams provided by the Government, it became apparent that differently encrypted traffic (from distinct security enclaves) is multiplexed in a router between the end systems and the Data Communications Equipment (DCE). Finally, the necessary coding order of Compression -> Encryption -> FEC, in conjunction with the topology just noted, implies a red-black boundary effectively at Layer 3 and a partition between ATC compression and FEC functions.

#### 3.2 Classification.

The Application Traffic Controller must distinguish amongst different types of traffic flows, and potentially amongst multiple flows of the same type. The process of inspecting traffic to determine its type and identity is *classification*. This usage of the word should *not* be confused with security classification! This is a much more general usage.

Although classification can, in principle, be performed at any of several protocol stack layers, it is most typically done at Layer 3. Common techniques inspect the IP source and destination addresses, the Transport (OSI Layer 4) protocol identifier, and for common protocols (TCP, UDP, perhaps ICMP) the protocol source and destination ports. IP Precedence and Type Of Service (TOS) bits are also easily inspected. Other Layer 2-4 protocol header fields can, with varying degrees of support from COTS tools, be automatically inspected. This is partially defeated if the packet payload is encrypted: for instance, secure IP in IP tunneling not only hides the Transport protocol port number, but the very protocol ID itself as well (replacing it, in the outer IP header, with the value for IP in IP tunnels).

Therefore classification must be performed by an ATC subsystem in each security enclave. Processing based upon the classification should also, to the extent feasible, be performed there. Processing which inherently must manipulate the multiplexed stream of traffic from all the different enclaves, must be dependent only upon fields that remain visible at that point: fields of the outermost, thus unencrypted, IP header. To enable this, the classifiers in each security enclave must set bits in the IP header to flag any QoS attributes derived from inner headers, and the network encryption devices must copy such bits from the inner (encrypted) to the outer (unencrypted) IP headers when encapsulating traffic (standard NES devices do this).

Some important determinations to be made based upon classification are: whether compression, encryption, FEC or other encoding has already been done; whether further encoding likely would yield significant gains or is otherwise required or desired; and whether further encoding could be transparently decoded by the receiver.

#### 3.3 Coding.

Compression, encryption and FEC must be encoded in that order at the sender, and decoded in reverse order at the receiver. Compression must precede encryption because properly encrypted data is incompressible. Both of these must precede FEC encoding in the sender, because in the receiver, decompression or decryption of corrupted (and not yet error corrected) data yields at best error propagation, and at worst both completely useless data and loss of sender-receiver synchronization. In the previously described Government specified network topology, this implies that primary responsibility for compression must reside above, and for FEC below, Layer 3. Compression must be performed in the end systems and/or the ATC subsystems in the different security enclaves. FEC must be performed in the ATC subsystem that processes the multiplexed stream of traffic from all the different enclaves, near the DCE. While the *basic* FEC, to deal primarily with *corrupted* blocks, must be performed at or below Layer 3, it is certainly permissible to employ *additional* FEC, to deal primarily with *missing* blocks, at or above Layer 3. This means that Multicast Dissemination Protocol (MDP, 4.9.2) can be used in conjunction with hardware or other lower layer FEC.

### 3.4 Packet Queuing.

Prioritization and queuing, in this project, is primarily an Application Layer issue, as that is where awareness of content resides, and bandwidth is so low that typically only a single data object should be transmitted at once. However, it is feasible to concurrently support multiple traffic flows, provided that each can tolerate the consequent low performance. COTS traffic classification tools have corresponding traffic control tools that can be used (5.8.2.3).

### 3.5 Concurrent Multipath Routing.

The Internet Protocol specifications allow packets from the same source to the same destination to take different end-to-end paths through the packet-switched network. Over relatively long time scales, this does in fact occur. However, over short time scales, it typically does not. Routers select a single best path from those available, and forward all packets matching the routing keys along that one path. They do this even if one or more alternative paths are just as good, and the selected path is congested.

A different mode of operation has been investigated, developed and demonstrated by CTI in work for the Air Force Research Laboratory and Air Mobility Command (AMC). AMC aircraft typically carry multiple radios, each of which provides only a low bandwidth link, but several of which can operate concurrently. Each of the different end-to-end paths through the network is dominated by its wireless first hop off the aircraft. The wireline ground network can be regarded as having zero latency, infinite bandwidth, zero loss rate and zero error rate, relative to the wireless hops; and its paths cannot possibly be congested by all the traffic that a single aircraft can generate or accept, even if all the wireless media available on the aircraft were used concurrently. Therefore that is exactly what is done.

Rather than select the single best path, the Intelligent Adaptive Communications Controller (IACC) on the aircraft concurrently routes traffic over all the links leaving the plane. A peer IACC reciprocally routes ground to air traffic.

Substantial performance improvements have been demonstrated. Speedup has ranged from negligible (where the link speeds are badly mismatched and the Transport protocols are confused by out-of-order delivery) to *supra-linear*! By 'supra-linear' (a term from the parallel processing community) is meant, faster than linear: 3 parallel paths more than 3 times as fast as one path. This surprising speedup is achieved when the use of multiple concurrent paths avoids congestion of a single low bandwidth link with both data and ACKs, thereby reducing latency seen by the Transport protocol, eliminating needless (indeed harmful) Transport protocol retransmission.

It is understood that the network topology and DCE configuration contemplated by the Navy does not correspond with the AMC situation. However, there are typically multiple antennas on the mast. If there are antennas not being used concurrently with the one designated for IP traffic, and if electromagnetic compatibility and other concerns do not preclude their concurrent use, then this offers an opportunity to increase the effective bandwidth usable by IP. Admittedly, this is not current CONOPS, and spectrum (especially satellite transponder spectrum) is a scarce resource, limiting the applicability of this approach; yet it is viable for high-priority communications with submarines, as each will consume multiple concurrent SATCOM channels only for a very brief period.

One case in which this could be demonstrated today is concurrent usage of UHF TACSAT and EHF, by a submarine equipped with both, in a region covered by both: it is assumed that all boats have UHF, and most regions are covered by UHF TACSAT; so availability of EHF gear and EHF satellite coverage are the limiting items. Another potential case is identified in "Submarine Requirements for ADNS", which refers to "multiple LDR SATCOM links" and "dynamic link selection": Concurrent Multipath Routing can be seen as a special case of dynamic link selection, where the link is chosen not on a session by session, but rather on a flow by flow (or even packet by packet) basis.

No effort has been expended in this project to pursue this opportunity, but if the Government indicates an interest, CTI can expand the scope of the investigation to address the topic, and is prepared to demonstrate implementations that could be deployed almost immediately.

## 4 Transport (OSI Layer 4).

### 4.1 Section Scope and Background.

This section examines the problems associated with computer network communication over disadvantaged links and what might be done at OSI Layer 4 to address these problems. It also considers some specific requirements of the SPAWAR environment. This system, like the majority of dispersed networks, including the Internet, is mostly dependent upon the services of the Transmission Control Protocol (TCP) to transfer data and information between nodes. For this reason, we begin with a lengthy discussion of what TCP is and what its problems are in this environment. Following a discussion of potential solutions to these problems, including the possibility of replacing TCP, there is a discussion of the shortcomings of TCP in providing support for the mission operational requirements, and what might be done to meet these requirements.

### 4.2 What TCP Is.

TCP is a connection-oriented, Layer 4 protocol that provides assured delivery of data or information. For readers of this report who come from other disciplines, almost every word in the preceding sentence needs a definition:

- By *assured delivery* is meant that TCP will notify (assure) the sender that a message has successfully reached the destination only after the receiver has acknowledged receipt.
- To be able to do this, TCP must maintain connections between both the sender and the receiver, thus it is *connection oriented*.
- A *protocol* is a formal statement prescribing a manner of behavior. In this sense, TCP is not a computer program, but rather a *modus operandi* for any program that attempts to communicate in an assured way and as a good citizen of the Internet.
- Saying that TCP is a *Layer 4* ("Transport") protocol simply means that the function that it performs, packaging and transporting data or information from an application on one computer to an application on another computer as if the applications were communicating within the same computer, belongs by definition at Layer 4 in the OSI model. The routes over which this communication travels, as well as the procedures and mechanisms involved in the transfer are transparent to Layer 4 (TCP), being performed at Layers 3, 2 and 1. "Layer 4" does not imply any real location in the computer or even, necessarily, the system design, but is, rather, merely an abstraction which helps some people organize their thoughts. The functionality of TCP, for instance, can reside in a data-gathering program, which, according to the OSI model, would be defined as being at Layer 7 ("Application"). In fact, TCP is usually built into the operating system.

To gain an appreciation for the good attributes of TCP, it is helpful to compare it to its sloppy brother, UDP (User Datagram Protocol). UDP, like TCP, is a Layer 4 protocol. Unlike TCP, however, UDP is connectionless and assumes no responsibility for successful delivery of the package. UDP is the electronic version of First-Class Mail; it sends the package on its way and forgets about it. If the package gets there, fine; if not, well, here comes the next one. The receiver gets what he gets; the sender has no idea if it got there or not. In fact, the sender doesn't know if the receiver was listening. There are protocols based on UDP which enhance the probability that the receiver, if listening, will get the package. These "reliable" UDP protocols (RUDP), in some cases, even mimic TCP, informing the sender of delivery status, via either acknowledgements (ACKs) or negative acknowledgements (NAKs). At this point it becomes difficult to distinguish UDP from TCP. In like manner, there are TCPs, which, in the face of severely disadvantaged or proscribed feedback links, abandon all effort of assurance. At this point it becomes difficult to distinguish TCP from UDP. But never mind - we are talking about "pure" TCP and UDP.

### 4.3 *Why TCP Is Used.*

#### 4.3.1 To Communicate, a Common Language is Needed.

When one elects to communicate, it is necessary to select a mode of communication (a language) that is shared by all anticipated communicants. In the world of computer-based communication there exist certain standards bodies that specify these common languages. TCP is one of these specified languages, and that is a first reason to use it.

#### 4.3.2 How Good is Good Enough?

Protocols have been approved which meet varying user requirements. Data and information transmission requirements may be stated in two dimensions: currency and integrity. As a rule, the more you relax requirements in either of the dimensions, the more you can gain in the other; if you over-specify one, you pay for it in the other.

##### 4.3.2.1 Currency.

Applications such as real-time two-way audio, equivalent to telephone conversations, require that data be processed in order and without significant delay. Such applications, however, do not require perfect integrity: a dropped word or two here and there does not seriously impair the conveyance of information; the pattern matching and context-guided abilities of the human brain can fill in the gaps and correct the mistakes in all but the most lossy and garbled of transmissions. For such applications, integrity is of lower importance; assurance is a by-product of normal conversation, not of the equipment over which it travels; accuracy in the spoken word has a much lower standard than, say, data fed to an automatic mission control system. Re-transmissions are handled at 'Layer 8' (radio operator says "Say again your last" or user simply asks "What?"). UDP is an appropriate protocol for this type of application.

##### 4.3.2.2 Integrity.

The major reason to use TCP is integrity of the delivered data. When a file is transmitted, it is usually important that all of it be delivered. Some incomplete files can be used (e.g., a list of addresses for a mass-mailing campaign, or uncompressed audio files), but complete and uncorrupted files are often needed (e.g., navigational way-points). For such files, the sender must know if they have been received successfully, or not. If not, the sender can try again. If the sender is ultimately unable to successfully transmit the file, it needs to be made aware of the failure so it can take other action or notify a human. So, in summary, you would use TCP instead of UDP when:

- the file must get there in its entirety, *and*,
- you need to know if it did, *and*,
- you need to know if it didn't.

You need a protocol that will try hard to get the message through, but report its failures. That's integrity. That's TCP.

### 4.4 *Roughly How TCP Works.*

TCP cannot "ensure" that the package will arrive at the destination any more than can FEDEX™, but, like FEDEX™, and unlike UDP or First-Class Mail, it does not simply send the package on its way assuming a perfect delivery channel. TCP (like FEDEX™) monitors transmission status, detecting failures and taking measures to correct them:

- Does it look like the receiver can't be reached? Give up and tell the sender.
- Is acknowledgement overdue? Re-transmit the package.
- Was I hasty? Wait longer next time.
- Is the receiver complaining about missing pieces? Send replacements (re-transmit).
- Am I overloading the system? Slow down.

The essence of TCP is that if it says the message got there, the sender can be assured that it did.



## 4.5 TCP Problems.

The reasons for using TCP are, to recapitulate:

- It is a universal standard.
- It is mature.
- It provides assured delivery.
- It behaves well on the Internet.

TCP, however, has a few problems in satellite and radio (wireless) environments. It was designed to work in a terrestrial environment, in which propagation delays are short, error rates are low, and bandwidth is generous. Assuming that lost packets must be caused by congestion in the network, TCP's response to loss is to slow down. Slowing down is inappropriate if the cause of loss is corruption. In addition, TCP's default assumptions concerning network characteristics cause it to perceive long delay as loss. A false perception of loss has two negative effects: first, a re-transmission of the 'lost' packet is instigated, causing unnecessary additional traffic on an already overburdened link, exacerbating the problem; second, TCP slows its transmission rate, also unnecessarily. This combination of slowing down and repeating previously received packets diminishes effective bandwidth drastically.

TCP does attempt to adjust to actual network conditions; given enough time, it can presumably adjust to almost any situation. In effect, however, this adjustment almost never gets done: the adjusting procedure is slow and the length of the data being transmitted is usually such that the session ends, limping, long before TCP has become adjusted to the environment. In addition, the algorithms used by TCP to adjust are rather fragile and can be confused by highly variable, unevenly distributed system effects. This confusion can result in non-productive behavior and even lock-up.

## 4.6 Dealing With The Problems.

The following is a discussion of two approaches to solving the problems of communicating over disadvantaged links. Sadly, only the first approach deals with the real problem - the disadvantages of the link. The other approach deals, for the most part, with the vulnerabilities of TCP.

### 4.6.1 Addressing The Low Bandwidth Problem - Multiple Paths (Layer 3).

In order to increase bandwidth for wireless communications, some wireless systems employ multiple radios in parallel. Procedures, usually at Layer 3 (Network), force the data over these multiple radio paths. If this traffic were over the Internet, this solution could be seen as bandwidth hogging, thus violating the fairness doctrine (discussed later), but we don't care (also discussed later). Although SPAWAR systems do not currently use this approach, it may be viable, considering the availability of multiple antennas on the masts and our previous work in this area (3.5).

As if the actual handicaps of wireless and satellite communication were not enough to give TCP a headache, multiple transmission paths add the insult of packets arriving at the receiver out of order, and, even more destructive, acknowledgements coming back to the sender out of order. One possible effect of out-of-order delivery is that a given TCP implementation might see out-of-order ACKs as duplicate ACKs, thus instigating unnecessary retransmissions from the point of the out-of-order ACK and possibly also causing TCP to revert to exponential backoff based on erroneously perceived round-trip time. This is only one example out of multiple possibilities; for instance, RFC 1323 states: "Reception of an old duplicate ACK segment at the transmitter...is likely to lock up the connection so that no further progress can be made, forcing an RST [reset] on the connection."

TCP assumes that if a receiver acknowledges receipt of packet 5, it has already received packets 1 through 4 - packets 1 through 4 are history. If it receives an acknowledgement for packet 3 after it has received acknowledgement for packets 4 and 5, it can become confused and petulant. What happens then depends on the particular implementation. In the extreme (reputedly not rare) case, it will cease transmitting and either close the connection or lock up. In a less severe reaction, it slows to a crawl.

## 4.6.2 Addressing The TCP Confusion Problem – Data Link Automatic Repeat reQuest (DL-ARQ) (Layer 2).

In an effort to insulate TCP from the vagaries of wireless communication discussed in Section 4.5, efforts have been made to move responsibility for assured delivery over the wireless link to Layer 2. In this approach, Layer 2 takes responsibility for the problematic wireless link, which is the source of the majority of transmission failures. TCP is still responsible for all other links, before and after the wireless link. To the degree that this removes the confusing influences of wireless communication from consideration by the easily derailed algorithms of TCP, it is a good thing. Another good effect of this approach is that DL-ARQ, having a smaller header overhead, can more efficiently work in smaller packet sizes than can TCP, which means that the response to loss of a packet is retransmission of fewer bytes, thus imposing less additional traffic when retransmitting. Likewise, DL-ARQ only retransmits packets over a single hop, rather than the entire end-to-end network path. There are, however, countervailing negative effects of DL-ARQ. Analysis of its benefits must include an assessment of when these negative effects negate the benefits.

### 4.6.2.1 It Disguises Loss as Latency.

When TCP, in effect, consigns a packet to Layer 2, and Layer 2 then has problems getting it through the radio, all the time taken for retransmission(s) (and the time taken to decide that retransmission is necessary) appears to TCP as mere latency - it sends it out and a long time later it gets an acknowledgement, with no indication that retransmission due to loss had been necessary. So what does TCP do? It slows down. Of course, it would have done this in the face of loss, anyway (see Section 4.5), but its reaction would have been more severe. So, it is hard to see where this negative effect of DL-ARQ could ever be so bad as to negate its benefits. But it might (see Section 4.6.2.1.2).

#### 4.6.2.1.1 It Amplifies Mean Latency.

The bogus perception of loss as latency is amplified somewhat in that, for instance, if the true latency is 1 second and the loss ratio is 10%, the mean latency perceived by TCP will be 1.1 seconds (actually more, considering waiting for time-out, overhead incurred re-formulating the transmission, etc.). This is not a major consideration, just a nudge in the wrong direction. As a matter of fact, it might even be beneficial that TCP sees a longer than actual latency, thus slowing down to accommodate conditions, because, from its perspective, it *does* take 1.1 seconds, on average, to accomplish the transfer and acknowledgement.

#### 4.6.2.1.2 It Amplifies Latency Variation.

As mentioned before, the TCP algorithms are disturbed by variability. They are most efficient where variability is zero, as may be the case in communications over dedicated links completely controlled by the enterprise at hand, even wireless links. By disguising loss as latency, DL-ARQ introduces variability to the TCP algorithms where none existed before. The degree of damage done is a function of many things, mainly the relative degree of variability introduced, and the degree of sensitivity of the algorithms, which can vary from implementation to implementation.

### 4.6.2.2 It Increases the Incidence of OOOD (Out-Of-Order Delivery).

As discussed in Section 4.6.1, out-of-order delivery of acknowledgements can confuse normal TCP. DL-ARQ *may*, but *need not*, guarantee in-order delivery of Network (Layer 3) packets: it sees each as a single event. This is independent of its guaranteed in-order delivery of Link (Layer 2) packets (typically segments of Network packets). Thus, if it has no problem with packets  $n-1$  and  $n+1$ , but must retransmit packet  $n$ ,  $n$  will be acknowledged after  $n+1$ . Further, DL-ARQ is concerned only with the single hop to the immediately next node (the other end of the radio link). It assumes that if a packet is acknowledged by the next node in line, it is a success, and reports it as such. Should the packet be lost farther down the line, or should the receiving end time out, the ultimate receiver will signal the loss with a duplicate ACK of the previous packet. This will become an out-of-order ACK and will confuse TCP. DL-ARQ thus can increase the incidence of OOOD, even over a single path; the effect is compounded if there are multiple paths.

## 4.7 *Dealing With The Side Effects of The Solutions.*

We will now look at ways to improve the situation, assuming use of multiple paths to multiply bandwidth<sup>1</sup>.

### 4.7.1 The Side Effects.

#### 4.7.1.1 The Increased Out-Of-Order Problem.

It is well known that out-of-order packets and acknowledgements can cause serious problems for TCP. In light of the numerous different implementations of the TCP protocol, which vary from OS to OS and even from release to release, it is difficult to predict in advance the exact nature and magnitude of the effects. Presumably, TCP must tolerate *some* out-of-orderedness; it *was*, after all, implemented to operate over the Internet, in which packets can take differing routes to their destinations, some routes being slower or more congested than others. But this may be a bad presumption: TCP might gag on just one out-of-order packet, but the event is rare enough that the occasional crash of TCP, manifesting itself as a loss of connection, is accepted by the user as just another mysterious failure.

Let's assume that in our hypothecated environment (wireless communications over multiple paths) the problem is common enough to cause serious degradation. What, if anything, can be done about it? Since we cannot control what is running in the endpoints of the network, we must look to inserting something between these endpoint TCPs and the wireless links to shield them from out-of-order packets.

Inserting routers between the endpoints and the wireless link will do nothing to help the situation because it changes nothing, except to add another hop, another point of failure, and more delay to the path. All TCP activity is merely forwarded, as is, back and forth. To address the problem, we need a pair of gateways which make possible the modification of the TCP dialogue, managing the wireless link as it can do best, and pretending to each endpoint to be the other endpoint, an endpoint which never sends out-of-order packets or acknowledgements.

Using a standard TCP in this gateway is not seen as a feasible means for suppressing out-of-order packets to the endpoints. First, a standard TCP is not so equipped as to be able to suppress them. Second, it would have similar, perhaps even worse, problems than the TCP in the endpoint. This would just be introducing another point of failure specifically vulnerable to the problem we are trying to solve.

What is needed is a substitute for standard TCP in this gateway - one which will not misbehave in the face of out-of-order packets and acknowledgements, and which will not pass them on to the endpoints. Interestingly, we have already installed such a gateway for a current research project - the SCPS-TP gateway (Space Communications Protocol Standards - Transport Protocol). We have evidence in the form of a tcpdump taken during a recent experiment that four out-of-order ACKs were encountered during the transmission of thirteen packets during a period of just over one minute, without crashing SCPS-TP; and the out-of-order ACKs were not passed on to the endpoint.

We suspect that it was SCPS-TP's unique feature, SNACK (Selective Negative Acknowledgement) that caused this good behavior. SNACK is an option that enables the communicants to continue transmitting packets to the end of the stream without backing up to retransmit all packets from the first point of loss. It continues to the end (limited by window size), remembering all of the holes, filling them in during a clean-up process. In maintaining this list of holes, we think, SCPS-TP culls it, removing holes that it has subsequently discovered (from out-of-order ACKs) to have been filled, while at the same time ignoring out-of-order ACKs that fall into a range that has no holes.

So, SCPS-TP gateways appear to solve the problem of out-of-order ACKs. More directed experimentation to verify this suspicion might be in order. It might be, though, that re-examination of our library of tcpdumps from previous experiments and exercises will provide sufficient substantiating evidence.

---

<sup>1</sup> Recent experiments show supra-linear speedup: data transmission appears to scale linearly with channel aggregation (no overhead or thrashing), and the increase in the rate of ACK returns seems to keep the pipe fuller.



#### 4.7.1.2 The Additional Opportunities for Confusing TCP.

Most of the problems of TCP in the SPAWAR environment stem from the induced behaviors driven by the various algorithms associated with the Internet fairness doctrine: slow start, congestion prediction, congestion avoidance and exponential back-off. Some improvement can probably be derived from tweaking a few of the parameters associated with these algorithms, such as Initial Round Trip Time (IRTT). Increasing IRTT to something commensurate with the actual round trip time (observed to be 23 seconds in recent experiments for the AFRL/IFGR project) will save several rounds (presumably three) of adaptation to the environment, thus lowering load on the link at the beginning of a transfer and shaving perhaps 20 seconds off of total transfer time. For a short, one packet (1500 byte) message, this would mean one packet transmitted rather than four. The same fundamental problem exists, however - the (incompetent) attempt to be fair. Tweaking parameters can never attain the desired efficiency.

#### 4.7.2 Solution Approaches.

##### 4.7.2.1 Tune Standard TCP in the Endpoints.

As discussed earlier, the inefficiencies experienced when TCP is confronted by loss or long delay can, to some extent, be ameliorated by tuning certain of standard TCP's parameters and by invoking certain of its optional features. Not all flavors of standard TCP, however, make available the same parameters for tuning and features for invoking. Moreover, we should adopt an attitude of hands off the end nodes, since they will not always be ours to fiddle with. In any case, the endpoints under SPAWAR control should be tuned.

##### 4.7.2.2 Create a Gateway Using Standard TCP.

If we build a gateway using standard TCP, we are given a measure of freedom in choosing the flavor of TCP which best meets SPAWAR requirements, in a box that "belongs" to us and can then be tweaked to a fare thee well. We still must, however, live with the fairness doctrine - no standard TCP will let you disable it. On the other hand, this, as well as the next approach, presents opportunities, discussed in Section 4.6, to introduce multiple channels, thus increasing available bandwidth, and DL-ARQ, thus insulating TCP from variability in link conditions.

##### 4.7.2.3 Use Something Other Than Standard TCP in The Endpoints.

A simpler solution from a system architecture perspective would be to forego the gateways and replace TCP in the endpoints. This would even offer efficiency benefits from the perspective of a first order analysis of processing overhead, data manipulation and movement, and path length.

As mentioned before, however, we may not always have the opportunity to construct endpoints to our liking. As just one example, consider accessing an intelligence database maintained by another command. An additional, although less important effect would be that non-mission-oriented access to the Internet by personnel would not derive the full benefit of system improvements. Another strike against this approach is that all applications on the endpoints, including GOTS and COTS applications, which involve data or information movement between nodes and which currently use TCP (most of them) would have to be modified<sup>2</sup>. Although there are third-party TCPs which can be installed in the OSs (at the kernel level) of endpoint machines (e.g., Mentat TCP and SCPS-TP), thus avoiding the problem of modifying applications, we are still faced with the problem of endpoints over which we have no control.

The situation might conceivably be improved some time in the future with additional, IETF-sanctioned, options in the TCP header by which a requesting endpoint could, for instance, pass an expected RTT to the responding endpoint for use during the requested connection. We consider it highly unlikely, however, that any such extensions would include what we consider to be the highest payoff modification - that of violating the Internet fairness doctrine. Nor should they.

<sup>2</sup> In the GOTS Delta Load, all but two of the applications are COTS, and *all* of them are Microsoft OS-based.

Gateways introduce many negative factors to a system, including additional points of failure, increased hardware and software maintenance requirements, additional equipment cost and increased requirements for equipment space and load factors. As discussed earlier, they also introduce extra processing overhead and path length.

So why do we favor gateways for this environment? The major reason is that we cannot control all endpoints. In coming to this view we have perhaps exaggerated the importance of access to endpoints not under SPAWAR control. We are not privy to the frequency or importance of submarine access to external C4ISR systems or ancillary data sources, nor to the importance ascribed to crew access to the Internet for e-mail, family home pages, etc. Nor is it our charter to evaluate these factors with respect to the negative factors of introducing gateways. Although we see gateways as being at least the short-term solution, we are open to suggestions from those with a better understanding of the requirements. Another argument in favor of gateways can be found in 5.3.1.1.

#### **4.7.2.4 Create a Gateway Using Something Other Than Standard TCP.**

If we decide to introduce a gateway into the system, we have access to a broader range of potential solutions. Among these are: non-standard TCPs and non-TCP approaches (see 4.9).

Many of the enhancements specifically designed to deal with the problems of satellite and wireless communication were originally proposed in IETF RFCs and have found their way into standard offerings, as they are applicable to the Internet in general. For example, scaling of TCP window size to increase the maximum from 64kB to 1GB was originally proposed to address the long latencies involved in satellite communications. The general problem addressed was that TCP could not keep such a "long" pipe full, if it was at the same time reasonably "fat", if it had to stop and wait for acknowledgements after only 65kB had been transmitted<sup>3</sup>. As terrestrial copper and fibre-optic pipes moved into the gigabits per second range of performance, the other side of the delay bandwidth product equation, bandwidth (a *really* fat pipe), created a similar problem even for relatively "short" terrestrial links, so scaling of TCP windows found its way into standard TCPs.

Modifications not found in standard TCPs are of more interest to us:

If the TCP gateway acts as a proxy for the receiving endpoint, spoofing acknowledgements immediately to the sending endpoint, the effect will be to keep the pipe full regardless of the TCP window size of the sender. In addition, the RTT, as perceived by the sender, will be more regular and the algorithms will function better. This independence from TCP window size settings is especially valuable given both the varying environmental and topological circumstances of submarine communications and the impracticality of fine-tuning endpoints in response.

Special mechanisms, for instance Selective Negative Acknowledgements (SNACK) found in SCPS-TP, for dealing with loss, are also of interest, but the greatest benefit to be gained from a non-standard TCP gateway is freedom from the fairness doctrine.

TCP can be replaced in the gateways by a protocol based on UDP, but which has assurance capabilities similar to those of TCP, when feedback is available, and which, in the absence of feedback, provides at least enhanced reliability (see 4.2 and 4.8). This will involve, however, protocol translation, i.e., it must appear to the TCP-using applications in the endpoints as TCP - ACKing, providing TCP packet headers, supporting the three-way handshake, etc. Should this approach be deemed potentially rewarding, existing gateway products that include applicable protocol translation capabilities will be reviewed and acquired for testing. If none are found to be appropriate, we will look for implementations of appropriate protocols for endpoints which we might be able to use to develop gateway implementations. If these investigations prove disappointing, we will assess the practicality of original development effort based on existing protocol specifications, and, if found to be practical, will undertake to perform same. In no case do we anticipate the design of new protocols.

---

<sup>3</sup> Until submarine systems get access to much fatter pipes, this issue is not of concern.

## **4.8 Additional Requirements.**

We have discussed the challenges faced by standard TCP in the SPAWAR environment and high-level solutions to these problems. This discussion was directed toward communications requirements for the transmission of messages and files between single endpoints employing two-way dialogues. In this section we broaden the investigation to address two additional identified requirements: multicast and EMCON (Emissions Control).

### **4.8.1 Multicast.**

Some transmissions need to be sent to multiple receivers. For instance, weather updates are needed by every mobile unit in, or transiting to or through, the subject area. It is inefficient to conduct a separate communication session with each individual submarine: it is a burden on the sending site and it will mean delays of weather updates to submarines toward the end of the list - in the SPAWAR environment, these delays could be significant and dangerous. Multicast protocols overcome this inefficiency by sending once to a "group" comprising multiple receivers. It is a restricted form of broadcast, much like the premium channels on cable TV. TCP is by definition and specification a unicast protocol. Multicast is necessary, so we must look elsewhere.

### **4.8.2 EMCON.**

During most of a typical mission, the receiving end will be under EMCON and will be unable to support connection-oriented communication. It will not be providing any feedback (e.g., ACKs) to the sender. Since feedback is a basic tenet of the TCP protocol specification, this mode of operation is unavailable over TCP. Some few TCP implementations are capable of providing service without feedback but when they do, they are reverting to what amounts to UDP, providing no reliability mechanisms to replace the lost assured service. We see such solutions as providing a fallback for worst-case scenarios. In the SPAWAR environment, EMCON is the norm and should not be addressed as an infrequent aberration but should, rather, be specifically provided for.

Some multicast protocols provide assurance, using various forms of ACK and/or NACK response management, such as fan-out, consolidation and suppression. Other multicast protocols, in addition to these approaches, can fall back to a mode in which they assume that no responses from the receivers will be forthcoming, depending, instead, on transmission redundancy to enhance the probability of successful reception by listeners. Whereas the former class of multicast protocols may eventually be found applicable to the SPAWAR program, for this investigation we have concentrated on the latter class, which addresses both requirements - multicast and EMCON. Two general approaches are taken:

#### **4.8.2.1 Multiple Transmissions.**

In this approach, the sender sends the message multiple times, hoping that at least one will get through. In conditions where some receivers are not under EMCON, some implementations will base their re-transmissions on feedback from them, assuming that they represent overall reception conditions.

In the case of the subject application, it might be possible that all intended receivers are operating under EMCON. In this situation the sender would have no feedback providing information about overall reception conditions. To address this deficiency it might be useful to have equipment (radios, antennae, etc.), equivalent to that used by the fleet, at fixed locations, in areas representative of the operational regions, which could act as surrogates not under EMCON that would provide the necessary feedback concerning conditions. Thus, if a surrogate station on Diego Garcia were experiencing problems, it could be induced that submarines in the Indian Ocean might be experiencing similar problems and that re-transmissions in response to feedback from the fixed station would have a tendency to at least partially remedy losses experienced by the fleet, especially those caused by general disturbances such as sunspot activity or jamming. The benefit of this approach will vary with the areal generality of the cause of loss and the displacement of the submarine from the fixed stations.

#### 4.8.2.2 Forward Error Correction (FEC).

FEC encoding providing 'k of n' capability is considered more efficient and effective than multiple retransmissions. Efficiency may be measured in terms of how much of the resources are used to transfer the information. Effectiveness may be measured in terms of how fast the communication arrives at its destination. The following discussion is a surface analysis presented to illustrate the principle, not to prove the point... Suppose that it is required that a file be transmitted four times to achieve a 99% probability that it is received. If the file is 25kB long, this represents 100kB of traffic over the link. Time before receipt may be roughly calculated for a 1kB link --

1st transmission received - 25 seconds  
 2nd transmission received - 50 seconds  
 3rd transmission received - 75 seconds  
 4th transmission received - 100 seconds  
 The average (expected) time to receive is 62.5 seconds.

Now consider a hypothetical 'k of n' FEC encoding scheme. From the file, 10 FEC encoded pieces of 5kB each are created, any 5 of which can faithfully reproduce the file. This represents 50kB of traffic over the link. Time before receipt may be roughly calculated as follows for a 1kB link --

Best case - The first 5 pieces are received successfully - 25 seconds  
 Worst case - The fifth successful receipt does not occur until the tenth transmission - 50 seconds  
 The average (expected) time to receive is 37.5 seconds.

Comparing the two methods, it is clear that FEC 'k of n' encoding is both more efficient and more effective --

	Load on the Link	Expected Time until Receipt
multiple transmissions	100kB	62.5 seconds
'k of n' FEC	50kB	37.5 seconds

For these reasons we have limited reliable multicast selection to those protocols which exploit FEC.

### 4.9 Looking For Solutions.

We foresee a system based on gateways containing reliable multicast services and TCP services optimized for disadvantaged links. The survey of technologies, therefore, comprises three sets of non-exclusive criteria, the first for TCP, the second and third for non-TCP solutions. The TCP criteria are associated with performance, integrity and quality of service. The non-TCP criteria are associated with multicast and EMCON support. The sub-criteria for each of these categories may be seen in the table below.

#### 4.9.1 TCP Candidates.

The TCP candidates shown in the table are, for the most part, fallouts of our previous investigation on the AFRL/IFGR program. A summary of that investigation may be seen in the tables in Section 0. The versions of TCP selected for this investigation (not all of them from the previous work) are:

- Standard TCP - for a baseline comparison.
- SCPS-TP - Space Communications Protocol Standards - Transport Protocol.
- Mentat TCP - A commercially available TCP with advanced features.
- STP - Satellite Transport Protocol.
- T/TCP - TCP for Transactions.

## 4.9.2 Reliable Multicast Candidates.

As discussed in Section 4.2, the OSI Layer model is malleable when applied to specific system designs. Just as it is possible to instantiate the functionality of TCP in a Layer 7 application, one can view the transport services of reliable multicast protocols, such as MDP, to be provided at Layer 7, employing the lower-level services of UDP at Layer 4.

We, however, regard MDP and its ilk as Layer 4 services in that they provide general-purpose transport mechanisms, which can be employed by applications and end-users to satisfy specific transport requirements not addressed by either UDP or TCP. For this reason, we include this survey of reliable multicast protocols in the Layer 4 section of this report.

The major starting points for this survey were two previous surveys, one performed by TASC and the other from the University of Southern California published by the IEEE Communications Magazine. Between them 23 unique protocols were identified, of which three have been selected which seem promising for the application. From the following list, for various reasons, all were dropped from further consideration, except those indicated (in bold) as selected for further study.

- Adaptive File Distribution Protocol (AFDP)
- Isis Toolkit IPMC Transport
- Local Group based Multicast Protocol (LGMP)
- Log-Based Receiver-Reliable Multicast (LBRM)
- **Multicast Dissemination Protocol (MDP)**
- Multicast File Transfer Protocol (MFTP)
- Multicast Transport Protocol (MTP and MTP-2)
- Real-Time Transport Protocol (RTP)
- Reliable Adaptive Multicast Protocol (RAMP)
- Reliable Broadcast Protocol (RBP)
- **Reliable Multicast data Distribution Protocol (RMDP)**
- Reliable Multicast Framework (RMF)
- Reliable Multicast Transport (RMP)
- Reliable Multicast Transport Protocol (RMTP) (Lucent/AT&T)
- Reliable Multicast Transport Protocol (RMTP) (NTT/IBM Japan)
- Scalable Reliable Multicast (SRM)
- Single Connection Emulation
- Structure- Oriented Resilient Multicast (STORM)
- Transport Protocol for Reliable Multicast (TRM)
- Tree-Based Multicast Transport Protocol (TMTP)
- Tree-based Reliable Multicast (TRAM)
- Uniform Reliable Group Communication Protocol (URGC)
- **Xpress Transport Protocol (XTP)**

We also investigated P\_Mul, but learned that: its priority support is only available when running it as an application; and MITRE comparatively evaluated it against MDPv2, and SPAWAR subsequently selected the latter; so we have dropped P\_Mul from further consideration.

The table below summarizes characteristics of the surveyed protocols.

	Standard TCP	SCPS-TP	MENTAT TCP	STP	T/TCP	XTP	MENTAT XTP	RMDP	MDPv2	P_MUL
<b>Performance/Integrity/QOS</b>										
Non-Fairness	N	Y	Y	Y	?	Y	Y	Y	Y	Y
Capability	Y	Y	Y	Y	?	Y	Y	N	N	N
Refinements for Disadvantaged Links	Y	Y	Y	Y	?	Y	Y	N	N	N
Assured/Reliable (A/R)	A	A	A	A	A	A/R	A/R	A/R	A/R	A/R
Priority Support	N	N	N	N	N	N	N	N	N	N[Y]
Compression	N	Y*	N	N	N	N	Y	N	N	Y
Handshake Avoidance	N	N	N	Y	Y	Y	Y	Y	Y	Y
Mobility Support	N	N	N	N	N	Y	Y	Y	N	N
Multicast/Broadcast	N	Y*	Y	N	N	Y	Y	Y	Y	Y
Multicast/Unicast Together	NA	TBD	Y*	NA	NA	Y	Y	Y	Y	Y
Feedback Implosion Protection	NA	TBD	N	NA	NA	N	N	Y	Y	N
Late Join/Leave	NA	TBD	N	NA	NA	Y	Y	Y	Y	N
Support for EMCON	N	Y	N	N	N	Y	Y	Y	Y	Y
FEC	NA	N	NA	NA	NA	N	N	Y	Y	Y*
EMCON/NON-EMCON Together	NA	N	NA	NA	NA	N	N	Y	Y	Y
Resume NON-EMCON	NA	N	NA	NA	NA	N	N	N	N	Y
Availability	C	G	C	E	E	E	C	E	G	G

C=COTS

G=GOTS

E= Experimental/ Research, but available

\*=Considered or planned for future

Our discussion of Layer 4 solutions to the EMCON requirement has ignored complications presented by assured delivery services at other layers. For instance, CPS (Configurable Protocol Stack), currently being tested on the AFRL/IFGR project, is a DL-ARQ solution at Layer 2 which requires feedback from the receivers; without feedback, CPS will stall. Running an EMCON-capable service atop CPS will not provide EMCON-conformant communications.

#### 4.10 Recommendations.

We recommend gateways comprising support for both TCP and reliable multicast protocols. The selected candidate for the TCP gateway is SCPS-TP. The selected candidate for the reliable multicast gateway is MDPv2. Dark horse candidates in both categories, respectively, are Mentat TCP and Mentat XTP. A deeper discussion of these potential replacements can be provided as necessary.

We also recommend that strong error detection be provided at the Transport layer, at least, if FEC is not (3.3): it is desired to avoid redundancy of FEC across layers of the protocol stack; but if there is not even error *detection* at the Transport layer, significant opportunities for system enhancement and optimization are precluded.

We lastly recommend investigation of the technology and CONOPS involved in supporting a 'quasi-EMCON' mode of operation, wherein traffic marked by the sender as critical, and *partially* received (sufficiently to determine that it was flagged as critical, but incompletely and/or with uncorrectable errors) would be NACKed. This recognizes that the accurate and complete reception of certain traffic may be more important than maintaining EMCON. Clearly, this is already a command option: a boat under EMCON may receive traffic, which upon delivery to the Captain, leads him to decide to come out of EMCON briefly (to ACK or NACK receipt, request retransmission, pull data suddenly deemed vital, or otherwise respond to the newly received information). The current proposal is merely to formalize and provide better software support for this existing option. Note that this mode would also require support at other layers of the protocol stack, especially Data Link (OSI Layer 2).



### 4.11 Background Information Concerning TCP.

The table below contains the results of research into the characteristics of various "flavors" of TCP (Transmission Control Protocol), a connection-oriented, assured delivery, Layer 4 protocol. This section is an update and re-work of previous research performed for the AFRL/IFGR program.

#### 4.11.1 TCP Characteristics Table.

	Standard TCP - 793	Boston	Reno-2001	Vegas	Tahoe	WINNT	WIN98	WIN2000	SCPS-TP	VTCP-1263	Linux 2.2.18	STP	New Reno	SACK TCP	FAK TCP
Object/Source Available? (O/S)	O		O	O	O	O	O	O	S		S	S		S	S
SACK - 2018	N	?	N	N	N		Y	Y	N		Y		YN	Y	Y
SNACK	N	N						N	Y	Y	N	?			
NAK - 1106	N		N	N				N	N		N	?			
FAK	N							N	N		N				Y
Delayed ACKs - 1122			Y		Y	Y		Y	Y		Y	N			
Slow Start Improvement - 2001	Y		Y	Y <sup>4</sup>	N	Y		Y	Y <sup>4</sup>		Y	?			
Scaled Windows - 1323			Y			N	Y	Y	Y		Y	Y			
PAWS - 1185-1323	Y					NY	Y		Y	?					
AIDA (FEC)		Y	N	N		NY	Y								
Congestion Prediction - 2001	Y		NY	Y		NY		N	Y			Y	Y		
Fast Re-trans. & Recovery - 2001	N		Y		N	N	YNY	Y	Y?		Y		Y	Y	N
Dead Gateway Detection - 816	N					Y		Y	Y						
TCP Header Compression - 1144	N					Y			Y <sup>5</sup>						
MTU Discovery - 1191	N					Y	Y	Y			Y				
Timestamps (RTT) - 1323						NY	Y	Y	Y		Y	?			
Best Effort Trans.								N	Y		N				
SWS Avoidance - 1122						Y		Y	Y		Y				
Keep-Alive Packets						Y		Y	N		Y				
Multicast						N	N	N	Y?		N				

Question marks or multiple entries in this table reflect incompleteness, inconsistency, ambiguity or vagueness in the available documentation. To the extent that any of these become significant issues, they will be resolved by testing with reference software implementations.

<sup>4</sup> VEGAS Variation

<sup>5</sup> Not same as 1144 (delta headers & go-back-n)

## TCP Reference Table.

This table identifies the referenced source upon which a "Y" or a "N" in the previous table was based.

	Standard TCP - 793	Boston	Reno	Vegas	Tahoe	WINNT	WIN98	WIN2000	SCPS-TP	VTCP-1263	Linux 2.2.18	STP	New Reno	SACK TCP	FAK TCP
Object/Source Available?			32	18	32			O			S	44		49,50	49
SACK		10	9	17	9		24	62			63		48,53		25,53
SNACK		10							8	4		44			
NAK												44			
FAK															25
ACK-ACK		10	17					62			63				
Delayed ACKs			32		32	11		62	8		63	44			
Slow Start Improvement	39		17,25	27		11		62	27		63	44			
Scaled Windows			39			11	50		8		63	44			
PAWS						50	24,50		8	4					
AIDA (FEC)		10	10	10		50	50								
Congestion Prediction	39		17,25,39	18,27		11		62	27		63	44	25		
Fast Re-transmit & Recovery	12		12,25		32,48	32	24,32,50	62	27		63		25	25	25
Dead Gateway Detection						11			26,27						
TCP Header Compression						11			8,27						
MTU Discovery						11,50	50	62			63				
Timestamps						50	50	62	27		63	44			
Best Effort Trans.									8						
SWS Avoidance						11		62	8		63				
Keep-Alive Packets						11		62	8						
Multicast															

## 4.12 References.

NOTE: all RFCs may be found via <http://www.rfc-editor.org>.

1. Improving TCP/IP Performance over Wireless Networks; Balakrishnan, Seshan, Amir and Katz; U. of California at Berkeley, NOV95; <http://HTTP.CS.Berkeley.EDU/~hari/papers/mcn.ps>
2. A Comparison of Mechanisms for Improving TCP Performance over Wireless Links; Balakrishnan, Padmanabhan, Seshan and Katz; U. of California at Berkeley; AUG96; <http://HTTP.CS.Berkeley.EDU/~hari/papers/ton.ps>
3. Path MTU Discovery [RFC1191]; Mogul and Deering; Stanford U.; NOV90;
4. TCP Extensions Considered Harmful [RFC1263]; O'Malley and Peterson; U. of Arizona; OCT91;
5. Enhancing End-to-End Performance of Information Services Over Ka-Band Global Satellite Networks; Bhasin, Glover, Ivancic and vonDeak; Lewis Research Center (NASA); DEC97
6. Strategy for Developing Expert-System-Based Internet Protocols; Ivancic; Lewis Research Center (NASA); MAY97
7. The Operation of TCP and UDP Protocols Over ATM Radio Links [LONGFISH/FITFEEL]; Scholz and Cassidy; DSTO Australia and Air Force Rome Laboratory; JAN95
8. MIL-STD-2045-44000, 12AUG96;
9. TCP Selective Acknowledgement Options [RFC2018]; Mathis, et al; Network Working Group; 1996;
10. Exploiting Redundancy for Timeliness in TCP Boston; Bestavros & Kim; Boston University; no date; [http://lite.ncstrl.org:3803/Dienst/UI/2.0/Describe/ncstrl.bu\\_cs%2f97-001](http://lite.ncstrl.org:3803/Dienst/UI/2.0/Describe/ncstrl.bu_cs%2f97-001)



11. Windows NT White Paper; *Dave MacDonald; Microsoft Corp.*; 1996  
<http://www.microsoft.com/ntserver/commserv/techdetails/techspecs/tcpip.asp>
12. Performance of TCP/IP Using ATM ABR and UBR Services over Satellite Networks; *Kalyanaraman et al; Ohio State University and Samsung Electronics*; 1996; <http://www.cis.ohio-state.edu/~jain/papers/satellit.htm>
13. TCP Selective Acknowledgements and UBR Drop Policies to Improve ATM-UBR Performance over Terrestrial and Satellite Networks; *Goyal et al; Ohio State University*; 1997; <http://www.cis.ohio-state.edu/~jain/papers/ic3n97.htm>
14. TCP/IP Performance over Satellite Links; *Partridge & Shepard; BBN Technologies*; 1997
15. Satellite ATM Networks: A Survey; *Akyildiz & Jeong; Georgia Institute of Technology*; 1997
16. Satellite ATM Network Architectural Considerations and TCP/IP Performance; *Kota & Jain; Lockheed Martin Telecommunications & Ohio State University*; 1997; <http://www.cis.ohio-state.edu/~jain/papers/kaband.htm>
17. TCP Vegas: New Techniques for Congestion Detection and Avoidance; *Brakmo et al, U. of Arizona*; no date; <http://netweb.usc.edu/yaxu/Vegas/Reference/vegas93.ps>
18. Advanced Protocol Design; *Brakmo et al; NSRG/U. of Arizona*; <http://www.cs.arizona.edu/protocols/>
19. Advanced Protocols for New Generation Internet; *Smirnov et al; HP Internet Philanthropic Initiative*; 1998; <http://www.neva.ru/pei98/hppr.html>
20. Internetworking ATM With DAMA-Controlled SATCOM; *Bivens et al; Rome Laboratory*; 1997
21. Internetworking ATM With Wireless Land Mobile (WLM) Communication Systems; *Bivens et al; Rome Laboratory*; 1997
22. Impact of Burst Errors on ATM Over Satellite – Analysis And Experimentation Results; *Kaltenschnee & Ramseier; Ascom Tech Ltd.*; 1995?
23. The FITFEEL Transmission Protocol; *Kennington; DSTO; Australia*; 1997
24. "Microsoft Ships Windows 98"; *Wayne Rash; Internet Week*; 22JUN98; pg.45
25. Analyzing the Performance of New TCP Extensions Over Satellite Links; *Christopher Hayes; Master's Thesis; Ohio University*; 1997; <http://jarok.cs.ohiou.edu/papers/hayes-thesis.ps>
26. TCP Extensions for Space Communication [aka SCPS-TP]; *Durst, Miller, Travis; MITRE and Gemini Ind.*; [http://now.cs.berkeley.edu/~gribble/summaries/glomop/tcp\\_space.html](http://now.cs.berkeley.edu/~gribble/summaries/glomop/tcp_space.html)
27. TCP Extensions for Space Communications (SCPS-TP); *Durst, Miller, Travis; MITRE & MCI*; 1996;
28. Internet Official Protocol Standards [RFC2400] ["STD1"]; *Postel, Reynolds; IAB; SEP98*;
29. Transmission Control Protocol - DARPA Internet Program - Protocol Specification; [RFC793]; *U. of Southern California; SEP91*;
30. Some Testing Tools for TCP Implementors [RFC2398]; *Parker & Schmechel, Sun Microsystems*; AUG98;
31. Space Communications Protocol Standards (SCPS) Integration into Satellite Operations Infrastructure; *R. Smith; AFRL*; JUN98; [http://www.rl.af.mil/div/IFB/sbir/list98/SBIR\\_list98-106.html](http://www.rl.af.mil/div/IFB/sbir/list98/SBIR_list98-106.html)
32. Automated Packet Trace Analysis of TCP Implementations; *V. Paxson; Lawrence Lab; U. of California at Berkeley*; 1997; <http://www.nrg.ee.lbl.gov/nrg-papers.html>
33. Requirements for Internet Hosts -- Communication Layers [RFC1122]; *R. Braden; IETF*; 1989;
34. "SCPS Reference Implementation version 1.1.1"; e-mail from *R. Durst*
35. "Why SCPS?";
36. "Some thoughts on the use of SCPS for Sea Dragon, Warriors' Internet"; memo; *B. Levitt; JPL*; 07JAN97
37. TCP Extensions for High Performance [RFC1323]; *Mathis, et al; Network Working Group*; 1996;
38. Performance of Common Data Communications Protocols Over Long Delay Links: An Experimental Examination; *Hans Kruse; Ohio University*; 1995; <http://jarok.cs.ohiou.edu/papers/>
39. TCP Performance over Satellite Links; *Allman, Hayes, Kruse, Osterman; Ohio University*; 1997; <http://jarok.cs.ohiou.edu/papers/>
40. An Application-Level Solution to TCP's Satellite Inefficiencies; *Allman, Kruse, Ostermann; Ohio University*; 1997?; <http://jarok.cs.ohiou.edu/papers/>
41. RFCs in HTML Format; *Freikamp & Bank; FH-Koeln*; <http://rfc.fh-koeln.de/doc/rfc/html/rfc.html>
42. DoD SCPS Strategic Plan; *G.M. Comparetto; MITRE*;
43. Characterizing End-To-End Performance; *VitalSigns Software*; <http://www.vitalsigns.com/products/vista/wp/characterizing.html>
44. Satellite Transport Protocol; <http://http.cs.berkeley.edu/~tomh/stp/stp-intro.html>

45. Orchestra: A Probing and Fault Injection Environment for Testing Protocol Implementations; U. of Michigan;  
<http://www.eecs.umich.edu/RTCL/projects/orchestra/overview.html>
46. TCP Trace; Ohio U.; <http://joarok.cs.ohiou.edu/software/tcptrace/tcptrace.html>
47. TRACELOOK; Ipsilon Networks; <http://www.ipsilon.com/~minshall/sw/tracelook/tracelook.html>
48. The NewReno Modification to TCP's Fast Recovery Algorithm; Floyd, Henderson; UC Berkeley; FEB99;  
<ftp://ftp.ietf.org/internet-drafts/draft-ietf-tcpimpl-newreno-02.txt>
49. Experimental TCP Implementations; PSC; Carnegie-Mellon U.; 17FEB99; <http://www.psc.edu/networking/tcp.html>
50. Enabling High Performance Data Transfers on Hosts; J. Mahdavi; PSC; Carnegie-Mellon;  
[http://www.psc.edu/networking/perf\\_tune.html](http://www.psc.edu/networking/perf_tune.html)
51. Satellite Communications in the Global Internet: Issues, Pitfalls, and Potential; Zhang, De Lucia, Ryu, Dao; Hughes Research Labs; 1997; [http://www.isoc.org/isoc/whatis/conferences/inet/97/proceedings/F5/F5\\_1.HTM](http://www.isoc.org/isoc/whatis/conferences/inet/97/proceedings/F5/F5_1.HTM)
52. Boosting TCP/IP Application Performance; Hutchinson; BMC Software;  
[http://www.bmc.com/Collateral/White\\_Paper/TACWP00A.fm/Boosting+TCP+IP+Application+Performance?DMW\\_FORMAT=html](http://www.bmc.com/Collateral/White_Paper/TACWP00A.fm/Boosting+TCP+IP+Application+Performance?DMW_FORMAT=html)
53. Ongoing TCP Research Related to Satellites; Allman, et al; IETF Draft;  
<http://www.ietf.org/internet-drafts/draft-ietf-tcpsat-res-issues-06.txt>
54. Enhancing TCP Over Satellite Channels Using Standard Mechanisms [RFC2488]; Allman, Glover, Sanchez; NASA Lewis;
55. Review of Multicast Protocols; © 2000 TASC, Inc. All Rights Reserved.  
<http://www.tascnets.com/newtascnets/Projects/Mist/Documents/Review.html>
56. INTERNET-DRAFT - P\_MUL; C. Riechmann; FGAN/FKIE; April, 1999  
<http://community.roxen.com/developers/idocs/drafts/draft-riechmann-multicast-emcon-00.html>
57. P\_Mul - Reliable Message Transfer Protocol for Multicast and EMCON Network Situations; Christian Riechmann; FGAN/FKIE; 1999; [http://www.fgan.de/~rie/pmul\\_transparencies.ps](http://www.fgan.de/~rie/pmul_transparencies.ps)
58. An X-400 Message Transfer Protocol for EMCON Network Situations; Christian Riechmann; FGAN; Jul97;  
<http://tonnant.itd.nrl.navy.mil/ipresearch/rm.html>
59. Multicast Dissemination Protocol version 2 (MDPv2); Joe Macker (NRL), Brian Adamson (Newlink);  
<http://manimac.itd.nrl.navy.mil/MDP/>
60. The Multicast Dissemination Protocol version 2 (MDPv2) Toolkit; <http://manimac.itd.nrl.navy.mil/MDP/MdpOverview.pdf>
61. Multicast Transport Protocols: A Survey and Taxonomy; *Katia Obraczka, Information Sciences Institute — University of Southern California* [http://www1.tlc.polito.it/~mellia/Other\\_papers/MULTICAST/Obraczka.pdf](http://www1.tlc.polito.it/~mellia/Other_papers/MULTICAST/Obraczka.pdf)
62. TCP/IP Implementation Details for Windows 2000 RC1; White Paper; *Dave MacDonald; Microsoft Corp.; Jul99*  
[http://www.microsoft.com/windows/server/Technical/networking/tcpip\\_implement.asp](http://www.microsoft.com/windows/server/Technical/networking/tcpip_implement.asp)
63. Linux Programmer's Manual TCP; 25 Apr 1999; (MAN TCP of Linux 2.2.18)
64. Linux Programmer's Manual IP; 11 May 1999; (MAN IP of Linux 2.2.18)
65. RMDP: an FEC-based Reliable Multicast protocol for Wireless environments; L. Rizzo, L. Vicisano; ACM Mobile Computer and Communication Review, v.2 n.2, April 1998); <http://www.iet.unipi.it/~luigi/mccr6.ps>
66. A Reliable Multicast data Distribution Protocol based on software FEC techniques (RMDP); L. Rizzo, L. Vicisano;  
<http://www.iet.unipi.it/~luigi/hpcs97.ps>
67. Xpress Transport Protocol Specification (XTP); XTP Forum; March 1, 1995;  
<ftp://dancer.ca.sandia.gov/pub/xtp4.0/xtp4.0-specification-2s.ps>
68. A Brief Introduction to the Xpress Transport Protocol; <http://www.ca.sandia.gov/xtp/xtpintro.html>
69. T/TCP: TCP for Transactions; Linux Gazette, issue 47; November 1999; Stacey, Nelson & Griffin; U. of Limerick;  
<http://www.linuxgazette.com/issue47/stacey.html>
70. Mentat XTP Datasheet; Mentat Inc.; <http://www.mentat.com/xtp/xtpdata.html>
71. Satellite Transport Protocol (STP): An SSCOP-based Transport Protocol for Datagram Satellite Networks; Tom Henderson and Randy Katz; U.C. Berkeley Electrical Engineering and Computer Science; WOSBIS '97; October 1, 1997; [http://www.cs.berkeley.edu/~tomh/talks/stp\\_wosbis.pdf](http://www.cs.berkeley.edu/~tomh/talks/stp_wosbis.pdf)
72. Session Layer Requirements for Multicast Internets; Rex Buddenberg; Naval Postgraduate School;  
[http://vislab-www.nps.navy.mil/~budden/lecture.notes/qos\\_paper.html](http://vislab-www.nps.navy.mil/~budden/lecture.notes/qos_paper.html)
73. Revision of MIL-STD-188-184..., Charles Gooding and David Aitken; SPAWAR Systems Center and Strategic Partnerships Intl.

## 5 Session / Presentation / Application (OSI Layers 5-7).

### 5.1 Section Scope and Background.

In the higher layers, an Application Traffic Controller (ATC) is required to provide traffic prioritization and queuing, cache management, compression services, and user feedback. The ATC will manage the utilization of the satellite link when available, queue traffic when the link is down, cache pull content and manage data compression. This is an essential function when connectivity is provided via an intermittent, low-bandwidth link. It provides transparent connectivity by shielding network clients from the lower level details of managing an intermittent link.

The ATC ideally should provide optimal QoS in an environment characterized by long latency, low data rate, high error rate, asymmetry (in the worst case, EMCON), and dynamic variation (worse, highly intermittent operation). These channel characteristics may preclude satisfaction of all ship<->shore communications push and pull requests. To ensure that mission critical traffic is delivered with adequate QoS, it may be necessary to give preferential treatment to specific types of traffic. At other times, circumstances may allow transmission of all levels of traffic. Outgoing messages and data requests will be queued until the submarine is able to communicate with the shore station; once a connection is established, messages will be disseminated according to the situation at hand.

Caching not only reduces bandwidth requirements, but also enables access to (potentially stale shipboard copies of) shore data while submerged. Compression, tuned to the applications, greatly reduces bandwidth requirements. Meaningful user feedback provides visibility into network queues, etc., enabling users to 'self-manage' their network usage, ameliorating user frustration, avoiding wastage of bandwidth with requests for transmission of data that cannot possibly be delivered within acceptable time windows (which would merely be discarded upon receipt), and supporting planning with realistic estimates of when critical messages and other traffic can be sent and received.

### 5.2 Compression.

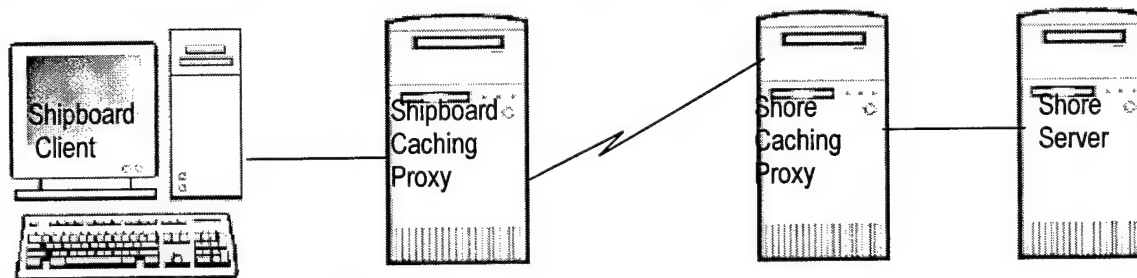
Compression performed at the Application Layer can exploit awareness of the format, content and intended use of the data. This enables high compression ratio techniques (both lossless and lossy) to be carefully selected and tuned, maximizing compression ratios, while ensuring that any resulting artifacts are unlikely to interfere with the application. Ideally, application software should be configured to perform compression and decompression without the need for involvement of the communications network or the users of data. Unfortunately, most applications do not incorporate compression. While it is possible for users to manually invoke utilities to decompress data upon receiving it and to compress data before sending it, such procedures would be a distraction from their primary duties and could result in the use of inappropriate techniques and/or specifications.

Therefore, an ATC should provide system administrators with the ability to specify rules that transparently redirect matching traffic through external compression routines prior to transmission, and correspondingly through decompression upon reception. Rules need to be written carefully – for instance, not all JPEGs are created equal: weather maps, typically, may be compressed further relative to their source format, with little consequence; but target area imagery intended for use by intelligence analysts may not be able to tolerate additional compression artifacts.

Compression interacts with error control. Image compression schemes that employ multi-resolution representations may imply a requirement for multi-layer FEC schemes, which more heavily protect the coarse resolution image layers. Such compression-driven FEC schemes fall within the general category of 'joint source and channel coding'. While their benefit is likely to be significant, they are not the focus of this effort: ATC compression research will focus on the general issues (rule based traffic matching and transparent rerouting through external compression and decompression engines), rather than on the details of compression algorithms for particular traffic types.

### 5.3 Caching.

Various caching strategies can be devised which enable off-line usage of data that is either static, slowly changing, or of a nature such that 'stale' data is still useful (most web content, some databases, etc.). This allows use of those resources while submerged, with opportunistic updating of the cached data being performed in a prioritized manner, during contact periods. The primary limitation on the use of caching is the need for *a priori* knowledge of what data to cache: not everything can be cached (due to timeliness requirements and storage capacity limits); but the data most likely to be needed during a patrol is not likely to be that most recently accessed while in port (as would be cached by typical algorithms). Despite this severe limitation, caching can provide significant benefit: for example, while submerged, sailors could browse issues of a home port newsletter, including images that would require significant bandwidth if they had to be transmitted over the radio link; absolute currency of such data is not essential to its general information and morale-boosting purposes. Procedures must be established to ensure that providing possibly stale data does not introduce risk: sometimes, even slightly outdated data cannot reliably meet the needs at hand, and may prove more detrimental than not providing the data at all. It is also important to support the 'push' of cache updates from the data source, when the source knows that use of the previous version might be detrimental.



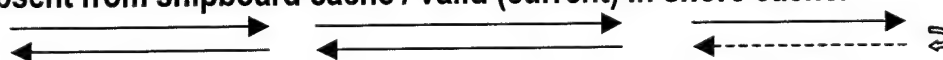
The scenarios shown below illustrate how the activity associated with satisfying a page request from the client varies depending on the status of the caches. Caching proxy and server requests/responses are dependent upon the absence or presence of data required to fulfill the request for a page and/or the currency of that data. Not all possible scenarios are explained here. Appendix B contains a full description of all tested scenarios and the results as reflected by each of the caching proxies. In addition to the presence and currency of data in the caching proxy, test scenarios consider the existence of a cache timer and the effect of timer expiration on the activity of the server and the caching proxies. In the illustrations below, solid lines represent requests from clients for data and replies from servers containing the requested data. Dashed lines from right to left indicate that cached data is current, while solid lines from right to left represent a new version of the data being sent to update a stale cached copy. It is assumed in these examples that the cache timer has expired in both gateways.

- **Item absent from both caches.**



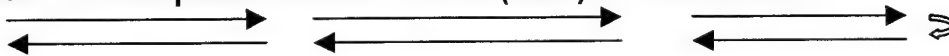
In the scenario shown above, the shipboard client sends a request for an item. The shipboard proxy intercepts the request, checks its cache for the item, does not find it, and forwards the request. The shore proxy then intercepts the request, checks its cache for the item, does not find it, and forwards the request. The server satisfies the request by sending the item, which is cached by both the shore and the shipboard proxies while on its way to the client.

- **Item absent from shipboard cache / valid (current) in shore cache.**



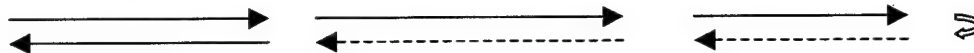
The client sends a request: the shipboard proxy does not find the item in its cache; but the shore proxy does, and sends a query to the server asking if its cached version is the latest. The server returns a message confirming that the cached version is indeed current, so the shore proxy returns its cached copy to the shipboard proxy and client.

- **Item absent from shipboard cache / invalid (stale) in shore cache.**

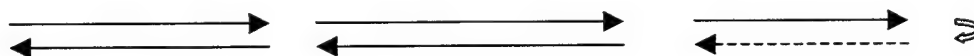


The shore proxy again finds the item in its cache and sends a query to the server to determine if its copy is current. The solid line from the server shows that, as the cached copy is stale, the server responds by sending the latest version, thereby updating the shore cache, populating the shipboard cache and satisfying the original client request.

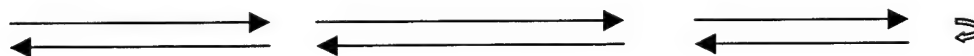
- **Item valid (current) in both caches.**



- **Item invalid (stale) in shipboard cache / valid (current) in shore cache.**



- **Item invalid (stale) in both caches.**



- **No ship <-> shore connection available to validate cache.**

More often than not, when data is requested, there will be no connection between ship and shore, so it will generally not be possible to determine immediately whether or not a cached copy contains the most current information. Discretion must be used in returning data that may not be current. Some applications may be tolerant of data whose currency cannot be validated. Other applications may be intolerant of the possibility of even slightly stale data.

While there exists no connection to the shore network, a shipboard client sends a request; the shipboard proxy intercepts the request, and checks its cache. If the item is not found, there is no decision to make -- the request is queued until a connection to the shore is available, and the user is both notified of when to expect a response, and given an opportunity to cancel the request. If the item is found, the proxy has 2 choices: satisfy the request with unvalidated cached data; or queue the request, just as if the item had not been found in cache, so the cached copy can be validated before returning it. The proxy must determine whether to return unvalidated cached data based upon the nature of the request. Below is a suggestion of categories of information and how each would be handled.

REQUEST TYPE	ACTION
GENERAL INFORMATION & ENTERTAINMENT: operational & technical manuals, regulations, forms; educational web sites, journals, papers; entertainment web sites, newsletters.	Send cached copy if one exists. If not in cache, queue request and notify user.
FREQUENTLY CHANGING CRITICAL DATA – e.g., coordinates of other military vessels, possibly enemy vessels.	Queue request until updated information can be retrieved. Old coordinates may mislead as the other vessel's intent is not necessarily known.
OUTDATED DATA THAT MAY BE USEFUL Such as weather maps.	Send cached copy with a timestamp so that the information can be put in perspective. Give the client the option of queuing the request for updated copy when connection is restored.



These categories are not hard and fast: they are examples of possible classifications, to be determined based upon approved doctrine and instantiated as rules by Navy system administrators. The example of the second category is somewhat contrived: this data probably really falls into the third category; but the point is that (perhaps) some data should not be used at all if its currency cannot be validated (for more discussion on this topic, see 5.3.1.2). Actions can also be contingent upon cached copy age.

HyperText Transfer Protocol (World Wide Web) makes requests and responses explicit, and was used in preliminary laboratory investigations, but various other protocols and services can be handled similarly. Microsoft Windows 2000 supports 'offline files': from a notebook PC, when connected to a server, a user selects a file and takes the action 'make available offline'; a local (cache) copy is made, which remains accessible even when the notebook is disconnected from the server; each time the user logs in or out, the local copy is synchronized with the server copy.

### 5.3.1 Other Considerations.

#### 5.3.1.1 Minimizing Burden on Wireless Link.

Particular attention should be given to design of the caching policies for each type of document, or even, perhaps, for each document. When at sea, requests for information that rarely, if ever, changes, should always be satisfied with a cached copy without even sending a request to the server to verify currency. For example, manuals of policies and procedures could be stored in the cache and used as if there were no wireless connection to shore thus freeing up valuable bandwidth for more urgent requests and more efficiently utilizing "up" time. One caching mechanism that can support this policy is to give a maximum expiration time to the document (RFC2068 places an upper limit of one year) such that it is never stale. In this case, for instance, requesting a cached page via the address field in Windows Internet Explorer will retrieve the page cached in the client without outside reference, thus imposing no load on wireless communications.

If the above page request, however, is made via the "refresh" button, the client does consult the next proxy device in the path, but stops there. In our notional system design, the next proxy device in the path is the "Onboard Caching Proxy". Since validation stops there if the proxy cache is current, no burden is placed on the wireless link. This is, by the way, another argument for having gateways (reference Section 4.7.2.3).

#### 5.3.1.2 Complications in Data/Information Acceptability Decisions.

As has been discussed above, relatively static information such as manuals can be used profitably even if they are not the latest release, and should not initiate expensive wireless communications to get the very latest while at sea. Other forms of data, such as geolocation data used, for instance, for a re-provisioning rendezvous, should be as fresh as possible; its expiration time should be short. In like manner, missiles should be launched with the freshest wind data available. Merely setting expiration time attributes, however, will not cover all contingencies inherent in these two operations.

In the case of a re-provisioning rendezvous, one could set the expiration time to 30 minutes. Should it happen that the time expires but a more recent fix cannot, for whatever reason, be retrieved, the old data is still better than no data at all. In the case of data concerning wind speeds and vectors at various altitudes, provided for missile guidance purposes, if the data is stale and cannot, for whatever reason, be updated, the decision to use it is, to put it mildly, complicated, guided by factors such as: target value, redundancy of coverage from other platforms, expected time until fresh wind data might be acquired, time criticality of the target, expected collateral damage of an off-course flight, anticipated military, political and moral consequences of expected collateral damage, relative value of preserving ordinance for use on other targets at a later time (when fresher wind data is available), etc.

Warning: most of the remainder of this section is based upon certain rash assumptions concerning Navy and, more specifically, submarine usage of intelligence data. It then incorporates popular fiction for exemplary purposes, and, near the end, might try the reader's patience with unwanted philosophical ruminations. If you wish to avoid these offenses, please skip to the next **bold** line.

Other examples of relatively static but, nevertheless changing information that can be useful even though not 100% current are S&T (scientific and technical data on Red and Blue equipment) and BIO (biographies of officers in opposing forces) data bases maintained by elements of the intelligence community.

Real-world examples of the usefulness of S&T and BIO data are hard to come by and, being had, would probably exceed the classification level (unclassified) of this report. If one can believe Tom Clancy, however (and who doesn't?), they can be of supreme importance. In "The Hunt for Red Oktober", Jack Ryan is (very messily) physically inserted into Capt. Mancuso's attack sub for the specific purpose of conveying S&T info (the newly developed and recently installed "caterpillar" drive on the Red Oktober), and BIO information (his rather intimate knowledge of the background (wife dead) and character (liberal, disillusioned) of the Red Oktober's commander). A remarkably happy confluence of situation-specific knowledge!

Although the dynamics of the plot dictated that the on-board S&T data base would have been stale ( the existence of the caterpillar drive was not discovered until after the boat sailed), the BIO data base would have been up to date had Ryan dutifully reported his insight to its keepers.

In neither case would stale data have been harmful. In both cases fresh data would potentially have been useful, even in the absence of the persuasive Mr. Ryan.

In theory, all of the complications discussed in this section can be addressed by reference to stated policy, doctrine and rules of engagement applied to mission objectives, and could therefore be automated. This body of guidance, of course, is not, and will never be complete or sufficiently correct to support full automation for an enterprise as important as that under discussion. The product of such a body of rules would be more than "data", more than "information", even more than "knowledge". It would have to be wisdom. Moreover, to automate this wisdom, we would have to trust the system which implements it, trusting that it was designed perfectly, implemented perfectly, and that it always operates perfectly.

We have indulged in this harangue to make a single point:

**There are situations for which no automated rules are sufficient, thus human intervention is required.**

**Corollary 1:** Data quality information must be supplied to the human operator.

**Corollary 2:** The human operator must have the means to communicate decisions to the system, overriding automatic procedures.

Simply put (finally), the decision to launch using stale data is a command decision and the computer is not the Captain.

## 5.4 Prioritization and Queuing.

The Traffic Prioritization and Queuing mechanism will allocate access to the satellite link using a weighted multi-level priority scheme based on user importance (rank and station), traffic importance (command email vs. personal email), mission need (an emergency evacuation request for example), and other relevant metrics. Some priority metrics would be fixed, such as who is captain of the boat, and others would be set by the users. An override capability will allow select personnel to lock out, filter, or delay traffic with priorities below a settable threshold. The queuing mechanism, upon link availability, will transmit the prioritized traffic in a manner that utilizes the channel most efficiently. An estimated transmit-time based on the priority level of each queued message and the statistical traffic history of recent satellite contacts will be calculated.

It is expected that the User Community will be accessible to provide input on desired prioritization metrics and their usage in the operational environment. It is ultimately envisioned that system administrators would be provided with a simple policy specification language and policy editor, which would enable them to define and modify prioritization policies based upon approved Navy doctrine (in commercial network management language, 'business rules'); these would then be interpreted by a policy enforcement engine. Initial investigations will use a *notional* prioritization hierarchy based upon seemingly reasonable *a priori* traffic classification and prioritization criteria.

Prioritization must be defined to provide for appropriate queuing and delivery of messages. Messages must be classified, based upon predefined factors, and then prioritized according to those classifications. The following section defines a notional set of classifications and a hierarchy based upon them. A sample scenario is provided to show how the classification and prioritization can be implemented using any predefined factors deemed appropriate.

### 5.4.1 Message Classifications And Hierarchy.

A hierarchy of classifications is suggested to provide the distinctions necessary for division of traffic in such a way as to provide appropriate data transmittal for any given situation. Within this suggested hierarchy, an initial division of traffic is made based upon a parameter or set of parameters. Further sub-classifications are defined based upon additional parameters and can address such things as urgency and relationship to the mission at hand. Distinctions would be made between operational and administrative traffic. The proposed approach includes a system of multiple queues, with further prioritization within the queues. Queue assignments will be based upon the defined hierarchy. Once a queue assignment has been determined, additional parameters will determine the forward prioritization assignment given to the messages within the queues.

As an example of how the proposed classification technique can be implemented, a sample scenario of message distinctions and corresponding priority designations is proposed in the following. Classifications defined in the sample hierarchy are a possible set of parameters for distinguishing traffic. These specific classifications and their corresponding priority assignments are arbitrary and can be shifted within the hierarchy or replaced with parameters deemed more appropriate based upon additional investigation into the situation.

In this example, the traffic is first assigned to one of six queues. Within these queues, traffic is then given a forwarding priority assignment determining which messages will be delivered first out of that queue. In addition to assigning forwarding position and priority, the traffic will be tagged with a "drop" priority to designate, should the queue become full, which messages will be dropped first. In addition to showing the possible hierarchy of queue assignments and priorities, suggestions as to how the messages will be identified as being in this category are given. The first division of traffic is made by distinguishing Navy traffic from Personal traffic. Within these two divisions, further distinctions are made based upon urgency and relationship to the mission. Mapping between these operationally meaningful criteria and information that can be derived from accessible packet header fields is addressed in a later section, and will be a major focus of the next phase of research and development.



## NAVY

- Message is specifically related to Navy or mission requirements.
- Source and destination addressees are military personnel.

**URGENT:** Successful completion of duties relies on "as timely as possible" receipt of this message.

### LIFE-THREAT

- Lives could be endangered if message is not received within a given period of time
- Military code word in the subject or the contents?
- Originating from "head honcho" of the mission?
- ALL messages in this class take highest priority and
  - WILL preempt other messages of any other class
  - CANNOT be preempted by any other class
  - Dispersed in FIFO order

### NON-LIFE THREAT

- Mission could be jeopardized if message not received within a given period of time
- Military code word?
- Preemption rules as follows
  - CANNOT be preempted by any class other than URGENT\_LIFE -THREAT
  - CAN preempt any messages other than URGENT\_LIFE-THREAT
  - Dispersed in priority order as described below

### ROUTINE:

#### MISSION SPECIFIC

- Destination includes only a specific submarine or only those included in particular mission.
- Dispersed in priority order as described below.

#### OTHER

- Destination is to all military personnel
- Dispersed in priority order as described below.

## PERSONAL

- Message is unrelated to Navy or mission requirements.
- Originator is not Navy personnel
- Originator is Navy personnel but message is indicated as personal.

### URGENT

- Timely delivery of this message is required.

### OTHER

- No deadline for receipt of message.

### 5.4.2 Prioritization within the Classifications (forwarding priority).

Within the queues, *forwarding* priorities of 1-4 will be assigned with 1 being the highest priority and 4 being the lowest. Priority 1 will be delivered first and 4 will be delivered last. Determination of which priority will be based upon the following factors:

Sender rank\* – one of five  
 Size of message\* – one of three  
 Type of message\* – one of many  
 Mission specific – yes or no

\*Indicates an easily identifiable parameter for determining classification of the message. "Type of message" is considered easily identifiable, provided standard Transport Layer protocol ports are used.

Higher rank receives higher priority.

Smaller message size receives higher priority.

E-mail messages will receive the highest priority from among the application types.

RANK	SIZE	TYPE
1 – O7 and up	1 – Small	1 - E-mail
2 – O4-O6	2 – Medium	2 - Office Document
3 – O1-O3	3 – Large	3 - Web page
4 – E7-E9/W1-W4		
5 – E1-E6		

One of 4 priorities will be assigned based on the above ranks. Overall rank is determined by adding the individual ranks of each parameter and assigning as follows:

Priority 1 – Rank totals of 3-4

Priority 2 – Rank totals of 5-6

Priority 3 – Rank totals of 7-8

Priority 4 – Rank totals of 9-11

The above-defined parameters can be expanded to include more divisions within the categories or reduced to less, depending upon current needs. In addition, the desirable number of different priorities can be altered. Priority determination based upon rank totals would then be altered to properly reflect the parameters being used and the desired distribution amongst the priorities. Finally, note that this need not be a "strict" priority scheme: low priority traffic should be protected from starvation.

### 5.4.3 Queue Size (drop priority).

Queue size must be limited: if a boat has WAN connectivity only for several minutes every several hours, queues easily could grow to hold more data than could be transmitted in the next one or even several periods of connectivity. If a maximum total queue size were designated, based upon how much data can be transmitted during the next few connectivity periods, dynamic queuing could allow higher priority queues to "borrow" space from lower priority queues. This is essentially *drop* priority: if the total of all the queues is only allowed to grow to a specified size, and if queues attempt to grow beyond that bound, something must be dropped; lower drop priority traffic gets dropped first. Further, should the communications officer declare a critical situation, he could direct that only mission critical traffic be serviced, and that dequeuing (and perhaps enqueueing also) of lower priority queues be temporarily disabled.

## 5.5 *Implementation of Prioritization and Queuing.*

To implement the above prioritization and queuing scheme, we propose to utilize Linux Netfilter and traffic controller. IPtables is the system administrator's interface to the Netfilter classifier, by which we can filter and mark traffic to designate membership in specific classes. These traffic classes can then be queued and sent in an order defined by the rules of queue hierarchies. Traffic controller (tc), a tool found in iproute2 and supported within the Linux kernel, provides what is needed to implement a variety of queuing disciplines, including Class Based Queuing.

### 5.5.1 Linux Netfilter.

Netfilter provides a means of filtering traffic based upon information found in packet headers. Packets are matched against criteria specified in IPtables rules; matching packets are handled according to the targets of those rules. Dropping, modifying or rerouting the packet may be performed, based upon source and destination addresses, transport protocol, and (for common transport protocols) source and destination ports. For example, if traffic is only to be accepted from specific sources, a rule may specify that unless the address of origin is one of the acceptable ones, the packet must be dropped. Packet headers can be modified, "mangled" in Linux parlance; the Type Of Service (or DiffServ) byte, for instance, may be modified, to mark a packet for specific handling by routers further along the path. The Linux memory structure that contains the packet (the socket buffer) may also be modified: for instance, by setting a numerical forwarding mark, to designate the packet as belonging to a given class. The Linux traffic controller subsequently can use this mark to distinguish between different classes of traffic.

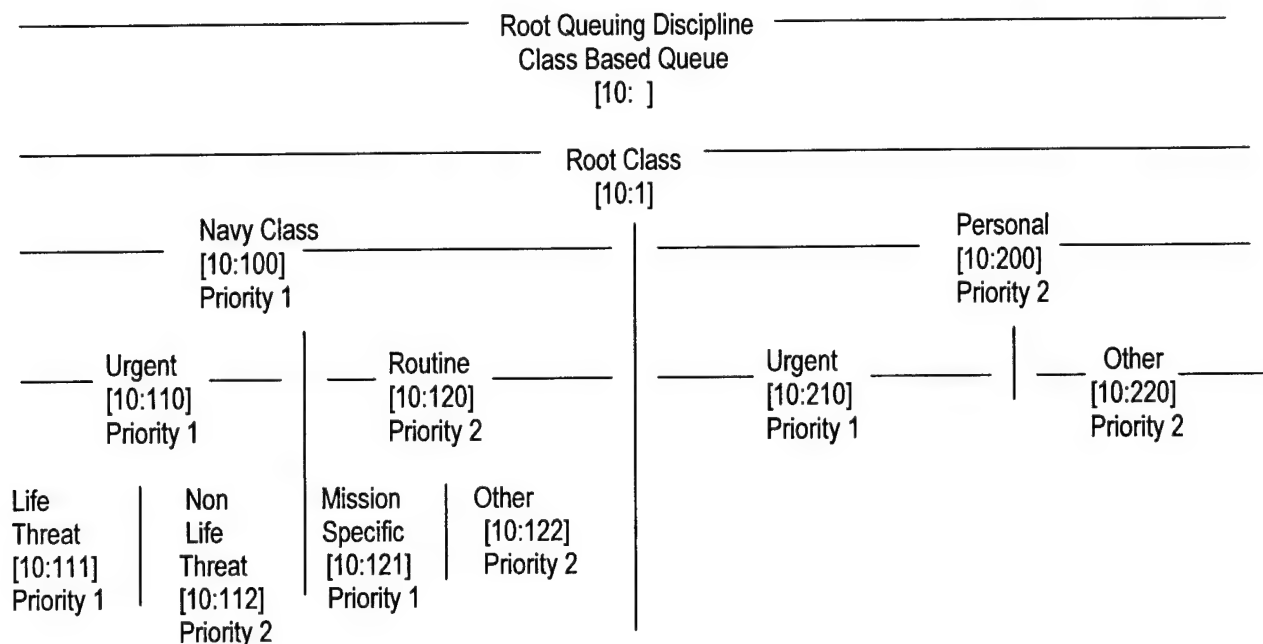
### 5.5.2 Linux traffic controller (tc).

Linux traffic controller (tc) is a user space program for controlling how outgoing packets are to be handled. It decides whether packets are to be queued or dropped and, if queued, the order in which they will be sent out. A queuing discipline must be defined for each output device. Nine types of queuing disciplines are supported, the simplest of which is a FIFO. The queuing discipline associated with each queue defines the way in which packets will be handled. Some of these disciplines use filters to classify the traffic and allow priority assignments based upon those classes. One such queue is the Class Based Queue. Class-based queuing disciplines may control one or more classes of traffic, each with its own queuing discipline, creating a hierarchy of rules by which traffic is separated, queued and sent out. Classes are assigned a class ID for identification purposes. This ID consists of a major number and a minor number. The major number (first number) corresponds to the queuing discipline the class resides in and the minor number identifies the class within that discipline.

### 5.5.3 Implementation of Priority Scheme.

In the following we will demonstrate just one way that these tools can be used to filter, classify and queue traffic in a manner that will achieve the objective. Mapping of the operationally meaningful traffic classes used below, to header fields easily accessible by Netfilter, can be shown, albeit in a rather contrived manner; development of more general, flexible and useful mapping techniques will be a focus of the next phase of research and development.

Since our priority scheme consists of a hierarchy of classes of traffic, the root queuing discipline of choice is the Class Based Queue (CBQ). A corresponding root class must be defined as well. We will assign our root class a class ID of 10:1, with the 10 corresponding to the root queuing discipline that this class resides in and the 1 being the root class identifier within the queuing discipline. As we have designated 6 different classes of traffic, we must create 6 classes within our root class, each with a corresponding queuing discipline. Within our root class we will define 2 subclasses of traffic, one corresponding to the Military class and the other corresponding to the Personal class. Each of these classes will "own" its own CBQ. Within the first class of traffic (Military 10:100), we will designate 2 more subclasses. Urgent traffic will be placed into one class, defined with a class ID of 10:110 and routine traffic will be placed into another class, which will be labeled with a class ID of 10:120. In the same manner, the Personal class of traffic will be subdivided into 2 more classes. The diagram below shows the class hierarchy along with the designated class ID labels. A priority is assigned to each of the classes designating the order of precedence for that particular group of traffic.



When classes 10:100 and 10:200 are set up, another CBQ is defined for each of them to allow for further classification. In addition, the same procedure is repeated for further classification within 10:110 and 10:120.

## 5.6 User interface.

User feedback will play an important role in managing the cached/queued system. Local shipboard email or HTML based feedback to the users' browsers will inform them of the status of their requests. For example, a user attempts to access a web page not currently in the boat's mirrored storage system. If the boat is currently at periscope depth and linked to the satellite, the queuing system will place the request in the queue and report to the user the estimated time until his request can be satisfied. Due to the low-bandwidth and intermittent nature of the satellite contact, the time to complete the transaction may fall several contact periods later for low-priority traffic. This type of feedback is essential for the user in order to mitigate the frustration experienced with low-bandwidth, intermittent connections, and to avoid wasting bandwidth by starting transfers that cannot possibly be completed within the time window.

A unique proposed enhancement is an 'enough button', which would enable an interactive user to halt consumption of wireless bandwidth by 'pull' traffic, as soon as he has seen enough of the response to satisfy his information need.

As demonstrated above, the tools existing within Linux allow for traffic filtering and queuing based upon a prioritization scheme. We propose to enhance this environment by providing a user interface that gives the ability to override default priority assignments and/or manipulate positioning of traffic within the queues. This interface would provide the user the ability to view his personal queue and, by selecting from pull-down menus, delete or modify the position of his own messages within this queue. In addition, the user will be informed of the position his messages hold within the larger queue overseen by the system administrator. This information would allow the user to see the results of his actions upon his own queue and how these may affect the result in the overall queue. The system administrator will also be able to modify or override precedence by moving things around within the main queue should circumstances make it necessary.

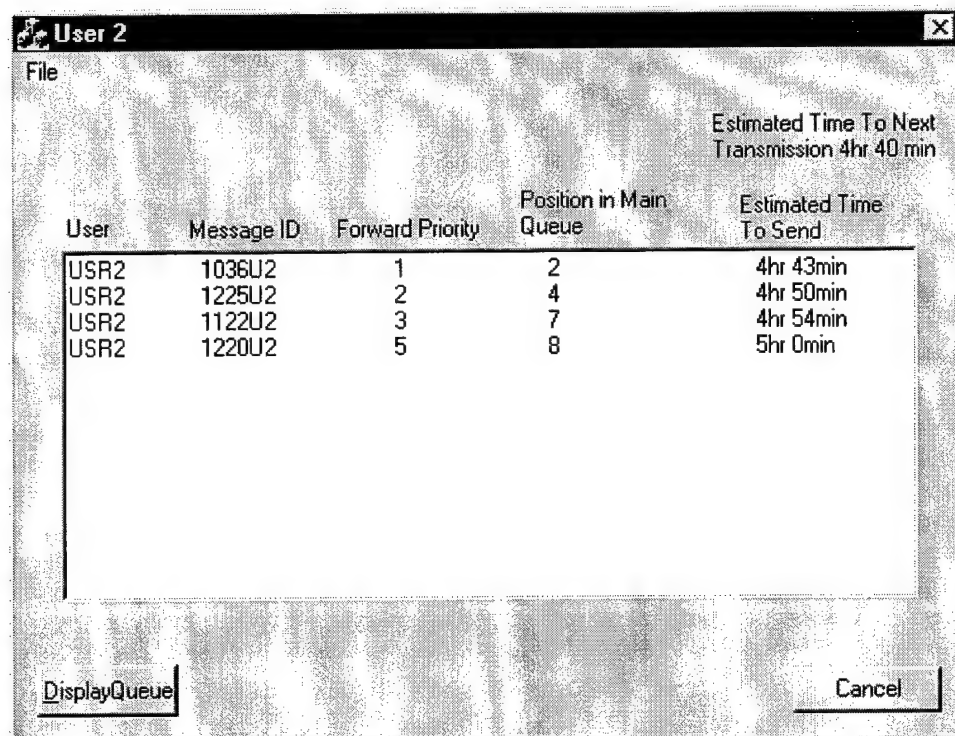
Information regarding the estimated time until the next messages will be sent is provided to those entitled to that information. If the message is being sent from a submerged submarine, the time would be the estimated time until the next surfacing. Messages originating on the shore would not necessarily be entitled to that information and therefore would not see it on the queue window. The opportunity to view and modify placement within the personal queue still exists, as does, for those so authorized, the ability to see the message placement within the larger queue.

The following screen shots will illustrate the functionality of this feature from both the user and the system administrator's point of view. In all examples the window does show the estimated time until the next transmission, although this information will only appear to users entitled to that information.

The screen shots in figures 1, 2 and 3 show the view of the queues as seen by User 1, User 2 and the System Administrator, respectively. The situation represents the queue status as a result of User 1 and User 2 sending messages and allowing them to fall as they may within the queuing system. For demonstration purposes, the scenario is oversimplified to consist of only 2 users and a system administrator. Screen shots 4 through 12 illustrate the results of modifications made by either user or the system administrator and the resulting view from the other perspectives. By seeing the results with just a couple of users, it can be inferred how much larger the impact of these changes would be in a system of many more users.

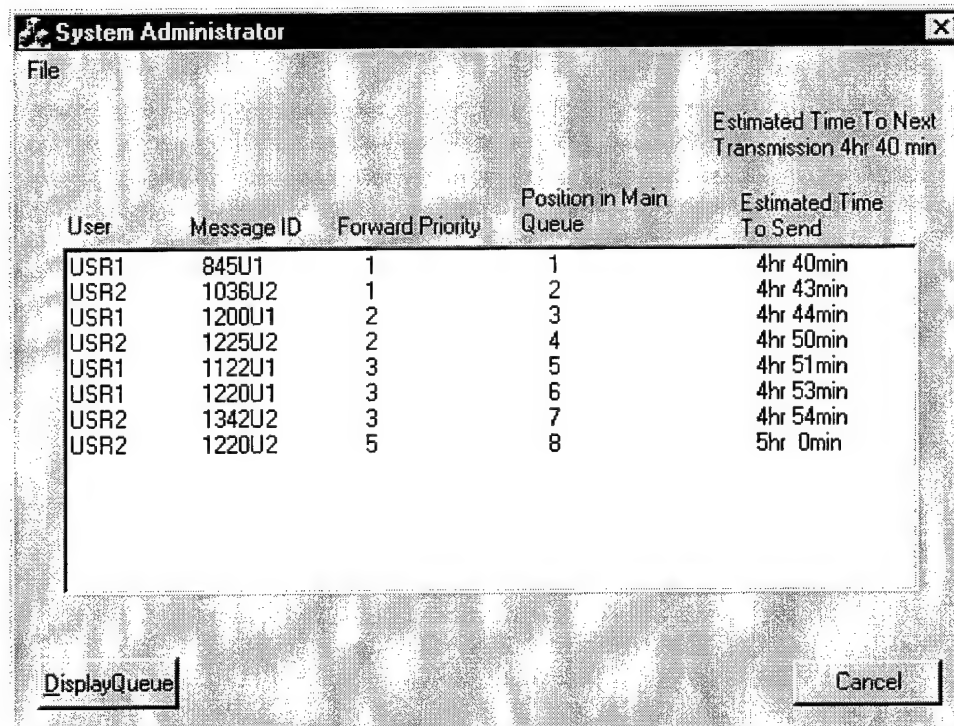
User	Message ID	Forward Priority	Position in Main Queue	Estimated Time To Send
USR1	0845U1	1	1	4hr 40min
USR1	1200U1	2	3	4hr 44min
USR1	1122U1	3	5	4hr 51min
USR1	1220U1	3	6	4hr 53min

FIGURE 1 – User 1 Default Queue



User	Message ID	Forward Priority	Position in Main Queue	Estimated Time To Send
USR2	1036U2	1	2	4hr 43min
USR2	1225U2	2	4	4hr 50min
USR2	1122U2	3	7	4hr 54min
USR2	1220U2	5	8	5hr 0min

FIGURE 2 – User 2 Default Queue

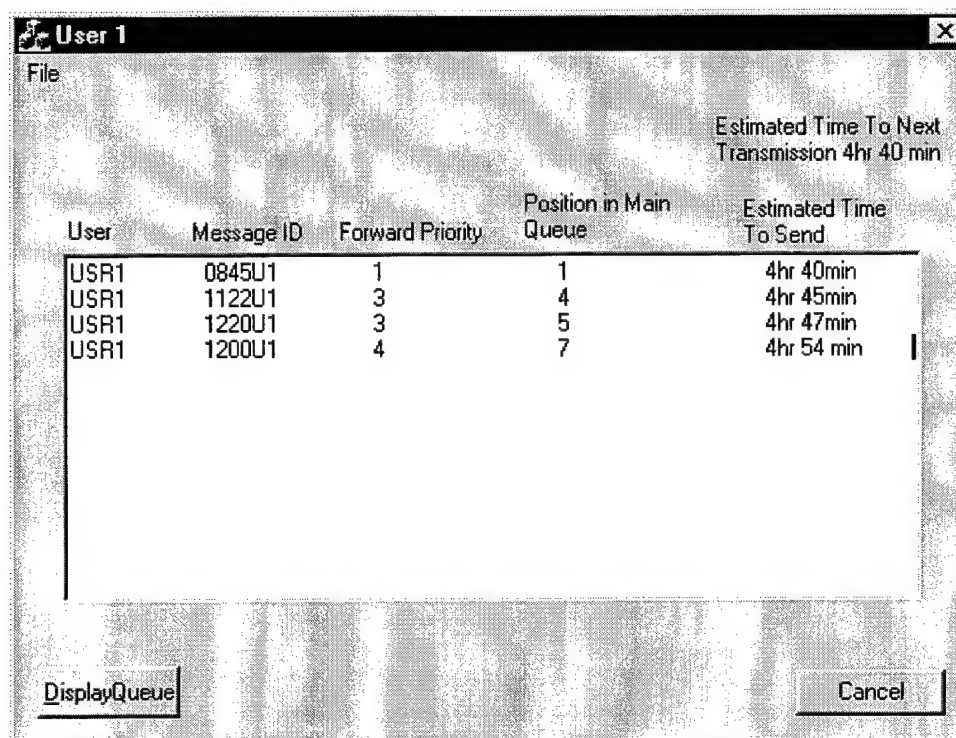


User	Message ID	Forward Priority	Position in Main Queue	Estimated Time To Send
USR1	845U1	1	1	4hr 40min
USR2	1036U2	1	2	4hr 43min
USR1	1200U1	2	3	4hr 44min
USR2	1225U2	2	4	4hr 50min
USR1	1122U1	3	5	4hr 51min
USR1	1220U1	3	6	4hr 53min
USR2	1342U2	3	7	4hr 54min
USR2	1220U2	5	8	5hr 0min

FIGURE 3 – System Administrator Default Queue

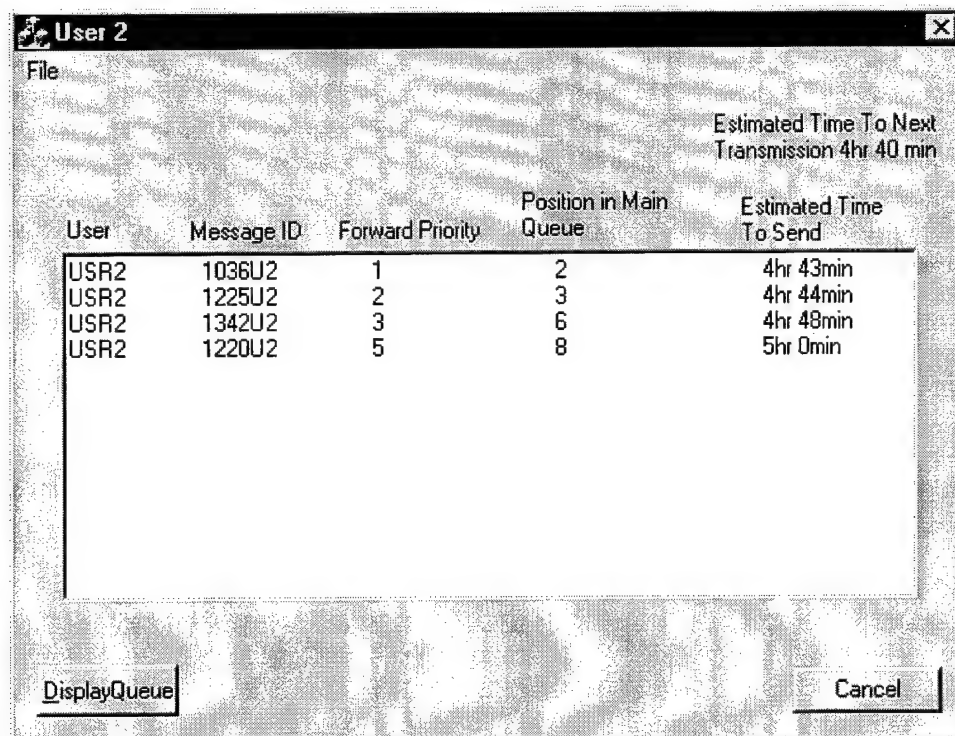


The next three figures illustrate the result on these same queues if User 1 decides to change a priority on one of his messages. User 1 changes the priority of his second message from 2 to 4, resulting in movement to last place within his personal queue. Figure 5 shows the perspective of User 2. Although he changed nothing in his own personal queue and all his messages remain in the same place relative to each other, his second and third messages have moved up one spot in the larger queue. Since the message that User 1 moved to lower priority happened to be a long transmission of about 7 minutes, a significant time gain is seen by User 2 even though his messages only moved up one spot. Figure 6 shows how this change affects the entire queue as seen by the System Administrator.



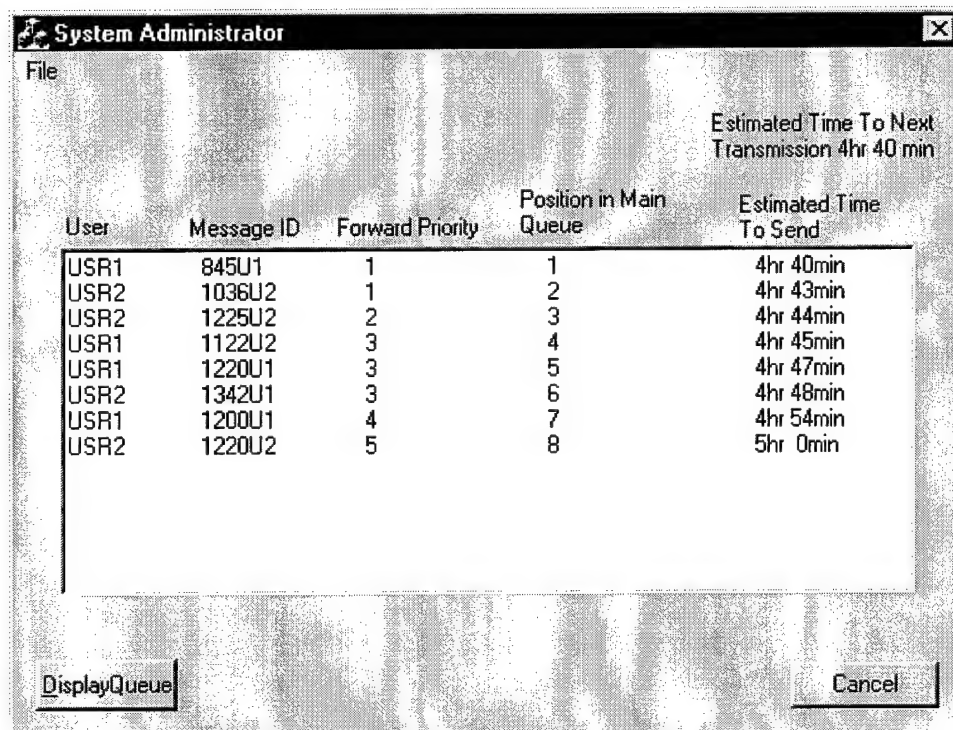
User	Message ID	Forward Priority	Position in Main Queue	Estimated Time To Send
USR1	0845U1	1	1	4hr 40min
USR1	1122U1	3	4	4hr 45min
USR1	1220U1	3	5	4hr 47min
USR1	1200U1	4	7	4hr 54 min

FIGURE 4 – User 1 priority change



User	Message ID	Forward Priority	Position in Main Queue	Estimated Time To Send
USR2	1036U2	1	2	4hr 43min
USR2	1225U2	2	3	4hr 44min
USR2	1342U2	3	6	4hr 48min
USR2	1220U2	5	8	5hr 0min

FIGURE 5 – User 1 priority change from User 2 perspective



User	Message ID	Forward Priority	Position in Main Queue	Estimated Time To Send
USR1	845U1	1	1	4hr 40min
USR2	1036U2	1	2	4hr 43min
USR2	1225U2	2	3	4hr 44min
USR1	1122U2	3	4	4hr 45min
USR1	1220U1	3	5	4hr 47min
USR2	1342U1	3	6	4hr 48min
USR1	1200U1	4	7	4hr 54min
USR2	1220U2	5	8	5hr 0min

FIGURE 6 – User 1 priority change from System Administrator point of view

In the next sequence of events, User 2 decides to delete one of his messages from the queue. Again, the result of his action is shown from the point of view of himself, User 1 and the System Administrator. The status of the queues already reflect the change previously made by User 1.

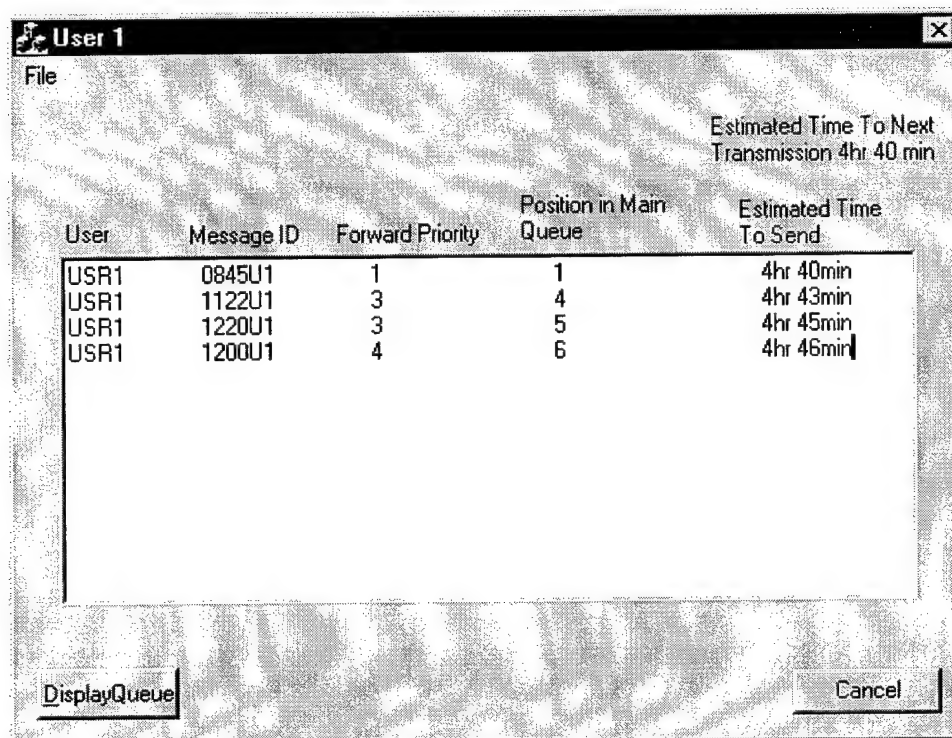
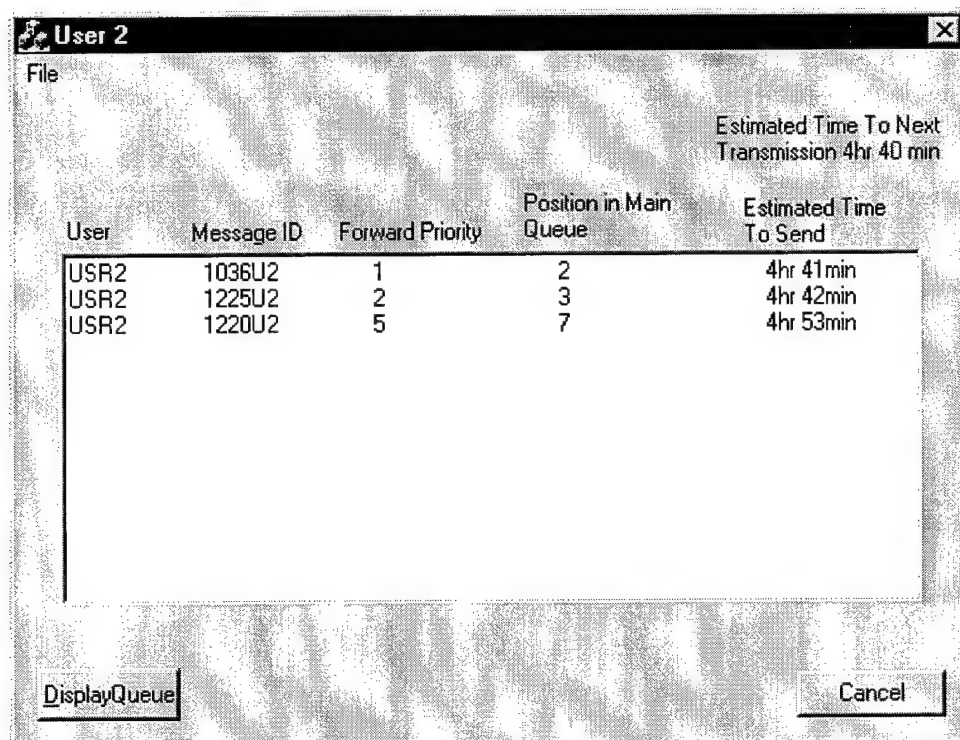


FIGURE 7 – User 1 view as result of User 2 deletion

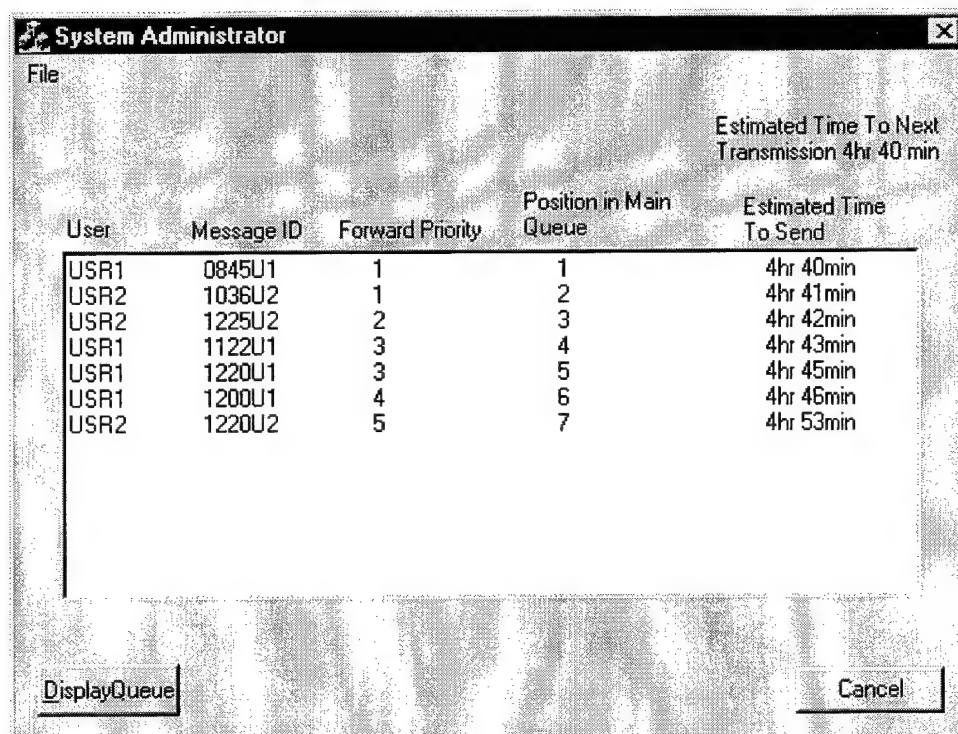


Estimated Time To Next Transmission 4hr 40 min

User	Message ID	Forward Priority	Position in Main Queue	Estimated Time To Send
USR2	1036U2	1	2	4hr 41min
USR2	1225U2	2	3	4hr 42min
USR2	1220U2	5	7	4hr 53min

DisplayQueue Cancel

FIGURE 8 – User 2 queue after deletion



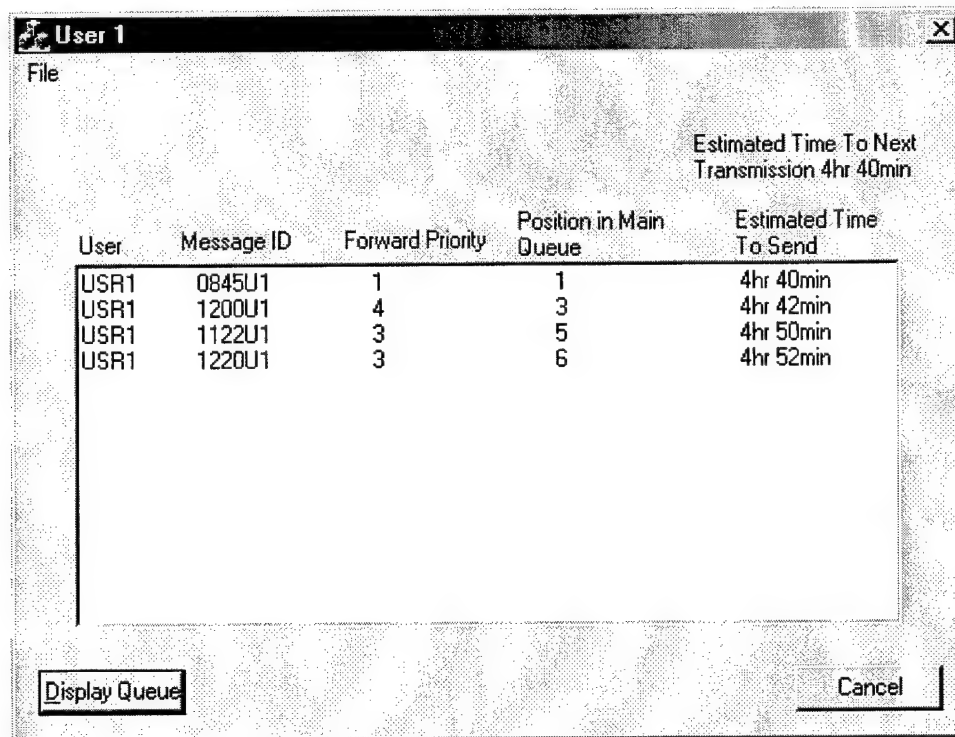
Estimated Time To Next Transmission 4hr 40 min

User	Message ID	Forward Priority	Position in Main Queue	Estimated Time To Send
USR1	0845U1	1	1	4hr 40min
USR2	1036U2	1	2	4hr 41min
USR2	1225U2	2	3	4hr 42min
USR1	1122U1	3	4	4hr 43min
USR1	1220U1	3	5	4hr 45min
USR1	1200U1	4	6	4hr 46min
USR2	1220U2	5	7	4hr 53min

DisplayQueue Cancel

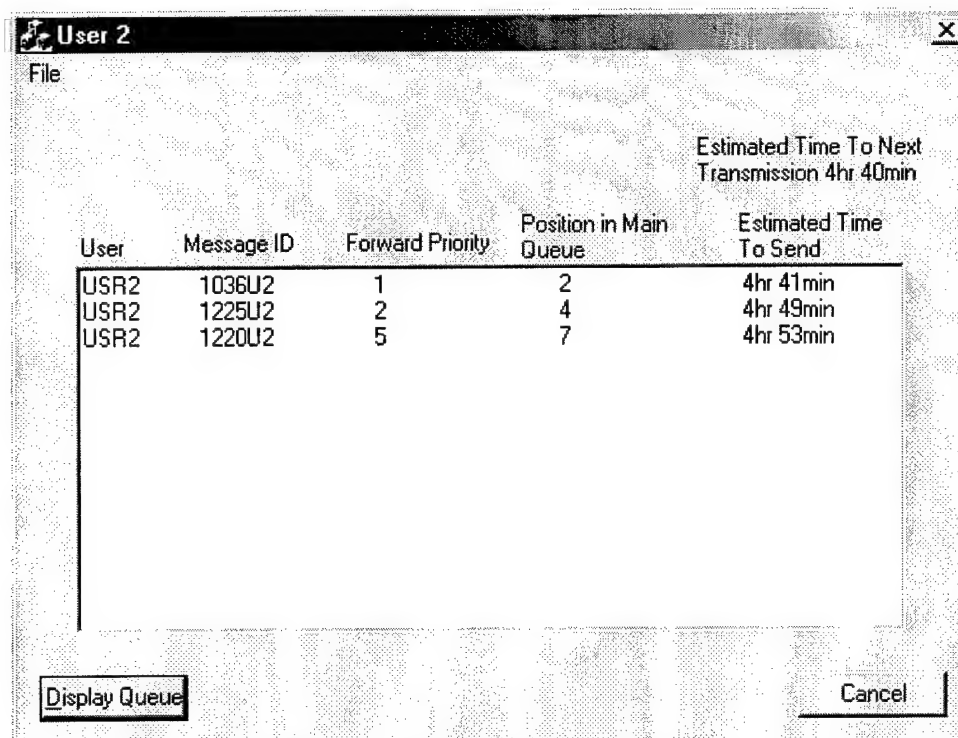
FIGURE 9 – System Administrator queue  
After User 2's deletion

As previously mentioned, in addition to allowing the user to view and modify his own personal queue, the System Administrator has the authority to override or modify assigned priorities if necessary. The last three figures illustrate this situation. In this example, the System Administrator has a need to move the message that currently occupies 7<sup>th</sup> place in the queue to 3<sup>rd</sup> place in the queue. Figure 10 shows how this change is viewed by User 1, figure 11 shows how this affects User 2's personal queue and his view and figure 12 shows the System Administrator queue after the message is relocated within the queue.



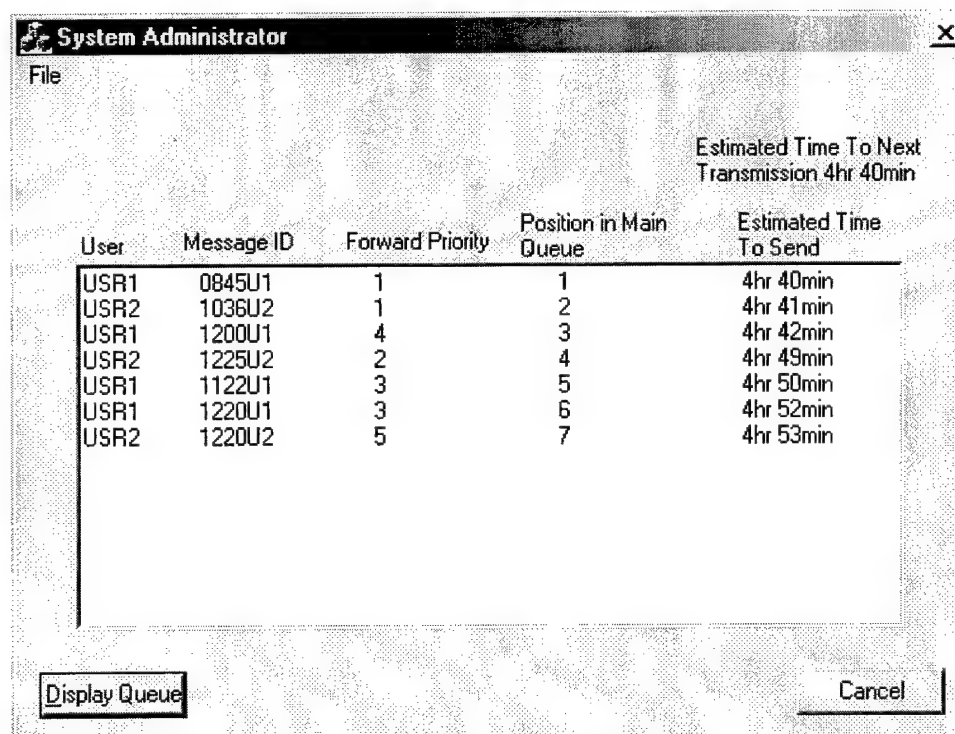
User	Message ID	Forward Priority	Position in Main Queue	Estimated Time To Send
USR1	0845U1	1	1	4hr 40min
USR1	1200U1	4	3	4hr 42min
USR1	1122U1	3	5	4hr 50min
USR1	1220U1	3	6	4hr 52min

FIGURE 10 – User 1 queue after System Administrator change



User	Message ID	Forward Priority	Position in Main Queue	Estimated Time To Send
USR2	1036U2	1	2	4hr 41min
USR2	1225U2	2	4	4hr 49min
USR2	1220U2	5	7	4hr 53min

FIGURE 11 – User 2 queue after System Administrator change



User	Message ID	Forward Priority	Position in Main Queue	Estimated Time To Send
USR1	0845U1	1	1	4hr 40min
USR2	1036U2	1	2	4hr 41min
USR1	1200U1	4	3	4hr 42min
USR2	1225U2	2	4	4hr 49min
USR1	1122U1	3	5	4hr 50min
USR1	1220U1	3	6	4hr 52min
USR2	1220U2	5	7	4hr 53min

FIGURE 12 – System Administrator queue After change



## 5.7 *Supplemental Support.*

In reviewing "Submarine Communications Assessment: End to End Assessment of Current and Future Submarine Communications Systems, Volume I, JUL99", and the associated PowerPoint slides, it has come to our attention that a rigorous analysis of communications traffic patterns for varying submarine operational scenarios has been performed, resulting in what are called "simultaneity diagrams". Although this study was performed for forward planning purposes, we see potential for operational uses.

- Online Reference

It might be useful to the system administrator who is organizing message queues, as described in Section 5.6, to have a side-by-side display of the simultaneity diagrams as reference for his decisions.

- Online Recommendations

If the source data collected for the creation of the simultaneity diagrams exists in machine-readable format, it might be used by a rule-based system to generate message ordering recommendations and present them to the system administrator for approval and/or modification.

- Automated Queuing

Machine-readable data could support automated queuing in situations where the system administrator is, for whatever reason, unable to perform the function, or, in a more advanced system, eliminate this part of his job description altogether, with, of course, appropriate provisions for oversight and override.

## 5.8 *Prototyping tools.*

### 5.8.1 Background.

If the Government exercises the SBIR Phase I option, an Application Traffic Controller proof-of-concept prototype will be developed and demonstrated. This prototype will be developed using COTS technologies and tools. Investigations have been conducted to verify the availability of open source COTS software suitable for integration into such a prototype: to minimize cost, it is desired that the prototype, insofar as feasible, not be developed as new code, but rather be assembled from existing software components. The Linux operating system is a prime candidate for implementing the ATC, as it offers the capabilities required to support the desired functionality, with open source. The ATC can be regarded as a Quality of Service (QoS) manager: QoS is a means of providing more predictable and reliable network service. The ATC will employ QoS tools and techniques to support the prioritization, queuing and bandwidth management functions required to serve its purpose. Recent Linux kernels have the ability to support QoS with advanced routing. Also, there are packages currently available to address caching strategies. Lastly, the traffic classification tools used to enable QoS can also be applied to select traffic for opportunistic compression.

#### 5.8.1.1 QoS Under Linux.

The QoS implementation in Linux supports the Differentiated Services (DiffServ) architecture, which addresses QoS primarily by controlling the flow of outgoing network traffic. Traffic shaping enables dedication of bandwidth, management of network congestion, and prioritization of traffic. DiffServ can ameliorate the effects of low bandwidth and long round trip times, especially in wireless environments where resources are severely limited. Linux traffic control is based upon four conceptual components: queuing disciplines, classes, filters, and policing. Each network device has a corresponding queuing discipline to control how outgoing packets are sent. At the very least, this is a first-in, first-out queue (FIFO). However, QoS introduces the idea of multiple queues to differentiate traffic flows.

Within a queuing discipline may exist classes, which serve as a mechanism for different routing behaviors based upon certain identifying criteria. Routing behaviors include bandwidth restrictions, priority mappings, or even tolerance for dropped packets. Filters can be used to map a packet to a certain class therefore ensuring it will be processed in a certain way. Finally, policing can be used to place restrictions on a traffic flow such as a rate limit.

#### **5.8.1.2 Proxy Servers under Linux.**

A proxy server is commonly used in situations where connections from the local LAN to the Internet are limited in some fashion. The proxy server's caching techniques can greatly reduce the utilization of precious bandwidth. A proxy server works by making requests "on behalf of" the client. The proxy server and the remote server then negotiate a transmission if the proxy's cache is not up to date. However, a small amount of traffic is still generated to make that check. Generally, this strategy is expected to reduce stress on the link. Investigation of caching strategies will focus on World Wide Web content: results easily can be generalized to other pull traffic; generalization to push traffic requires study. The Apache web server's mod\_proxy has been selected for initial caching experiments.

### **5.8.2 Experimentation in Feasibility.**

In order to verify that a Linux implementation would indeed be feasible, we have set up a small number of simplified experiments to verify Linux traffic control and caching as plausible foundations for a solution. Controlled experiments for both traffic control and web caching were conducted upon a simple testbed and later analyzed to determine how the hypothesized behavior correlated to the actual behavior.

#### **5.8.2.1 Testbed Configuration.**

A small testbed was constructed upon which our tests were executed. The testbed consisted of a mixed environment of both Windows and Linux with 4 machines in total.

- Machine A, a Microsoft Windows NT 4.0 Workstation to serve as a client for Web, FTP, etc.
- Machine B, a Microsoft Windows NT 4.0 Server acting as the remote Web and FTP server.
- Machine C, a RedHat Linux 7.0 gateway for the client side.
- Machine D, a RedHat Linux 7.0 gateway for the web server side.

It is important to note that the wired connection between C and D, although physically Ethernet, had its bandwidth throttled to roughly equate to a wireless link. The RedHat machines were upgraded to the most recent stable kernel, 2.4.7<sup>6</sup> and were configured to support the advanced routing, NetFilter, and QoS capabilities.

#### **5.8.2.2 Web Caching.**

Six tests were performed in the area of web caching. These controlled experiments tested two variables: reduced bandwidth (traffic control) and caching. Web caching has three possible states: not cached, cached and current, and cached but not current. A test was performed for each of these three states both with and without traffic control. Traffic control worked as expected effectively rate-limiting data transfer without having an adverse effect on TCP/IP (resets, retransmissions, etc). Web caching was also successful.

On the first request, the proxy pulled the entire GIF image since it did not already exist in the cache. On the second attempt, the proxy verified with the remote server that its cached entry was current and served it up to the client. For the third test, we made a change to the GIF image to force the cache to be outdated. The proxy, as expected, did another full transmission to acquire the latest version.

---

<sup>6</sup> Linux Kernel 2.4.x has been identified to have some problems with its TCP/IP functionality. However, this kernel is sufficient for our present testing purposes.

The proxy was then reconfigured for offline operation and the 'wireless' link broken. The proxy served items from cache without attempting to validate their currency with the unreachable server (0). Finally, a transparent proxy was installed on both Machine B and Machine C, IPtables rules were configured to redirect all HTTP traffic to the transparent proxy, and the Machine A browser was reconfigured to *not* explicitly use a proxy; the ability to transparently intercept traffic from non-proxy-aware clients and perform all the foregoing caching proxy operations was confirmed. Future work will extend this functionality to protocols other than HTTP.

### 5.8.2.3 Traffic Control.

Tests were performed to verify that network traffic could be reordered to transmit in a sequence other than that in which it would be transmitted by default. In our test setup, traffic was divided into two distinct and isolated classes, high and low priority. FTP traffic was given high priority while all other types of traffic were processed as low priority. The experiment was to initiate a HTTP web request first, and then begin a concurrent FTP GET, to see how the high priority FTP traffic integrated with the low priority HTTP data transfer. During the initial FTP control transmissions, traffic for FTP and HTTP was balanced quite equally. However, when the FTP session began to transmit actual data, the HTTP session was essentially halted until the FTP transfer had completed. The experiment was successful in demonstrating that Linux traffic control can effectively alter the order of packet output.

## 5.9 References.

- Linux Advanced Routing HOWTO (<http://www.linuxdoc.org/HOWTO/Adv-Routing-HOWTO.html>)
- Linux – Advanced Networking Overview (<http://qos.itc.ukans.edu/howto/>)
- Linux Traffic Control Implementation Overview
- Apache HTTP Server (<http://httpd.apache.org/>)
- Linux 2.4 Packet Filtering HOWTO  
(<http://netfilter.samba.org/unreliable-guides/packet-filtering-HOWTO/index.html>)
- Hypertext Transfer Protocol -- HTTP/1.1 [RFC2068]; IETF; Fielding, et al; JAN97;  
<http://www.w3.org/Protocols/rfc2068/rfc2068>
- Submarine Communications Assessment: End to End Assessment of Current and Future Submarine Communications Systems, Volume I; JUL99

## 6 Summary of Preliminary Conclusions.

The system's Physical (OSI Layer 1) channel characteristics and unique operational conditions have been identified. From that environmental information, Data Link (OSI Layer 2) requirements and desires have been derived. The Network (OSI Layer 3) gateway has been identified as the chokepoint for classifying all traffic for control by the ATC, and as the breakpoint between decoupled 'red' higher layer and 'black' lower layer ATC subsystems. The Transport (OSI Layer 4) has been partitioned into cases, based upon the interplay between the degree of need for reliability (assured delivery or not) and the operational mode (EMCON or not), and candidate protocols have been identified for each case (including two which might serve all cases).

A conceptual framework for prioritization and queuing has been drafted. Limitations of caching have been identified but the value of caching still appears more than adequate to justify its exploitation as one of a family of strategies for bandwidth management. A model for transparently inserting application-specific compression and decompression processes between applications and the network has been suggested. A user interface has been mocked up for the monitoring and management of Application (Layer 7) transmission queues.

The Linux operating system has proved itself to be a viable platform for developing an Application Traffic Controller. Web caching works well using Apache's mod\_proxy. In addition, there exists an offline mode, enabling the pull of pages from cache while off-boat links are unavailable (even to verify currency of the page, which must then be assumed, albeit with a caveat). Through the use of Linux traffic control, traffic can be prioritized on the basis of any information available in standard Data Link, Network and Transport Layer packet headers. If a one-to-one relationship exists between other information and packet header fields, additional criteria can be utilized. These empirically verified functions support development to fulfill all requirements for an Application Traffic Controller.

Linux based systems are at least as deployable as Windows based systems. When the Proteon routers are replaced, they could be replaced by Linux based systems with greater functionality and source code availability. Alternatively, it has been confirmed that the desired Layer 3 functions, although most flexibly configured under Linux, are largely available also under the OpenRoute software, and the higher layer functions can be decoupled from the Layer 3 functions; so there is no need to replace the Proteon routers to achieve the desired system functionality.

It appears feasible and appropriate now to proceed to the next phase of research and development, in which prototypes will be developed, and used to experiment with and demonstrate specific techniques for handling traffic, according to sysadmin-specifiable rules, reflecting doctrine and expressed in operational meaningful (rather than network technical) terms.

## Appendix A: Glossary of Acronyms and Terms

<b>ACK</b>	ACKnowledgement (typically of an assured transmission)
<b>ADNS</b>	Advanced Digital Network System
<b>ARP</b>	Address Resolution Protocol: maps addresses between Layers 2 and 3
<b>ARQ</b>	Automatic Retransmission reQuest: the basic assured delivery technique
<b>assured delivery</b>	confirmation (vs confidence) that transmissions are received successfully
<b>ATC</b>	Application Traffic Controller
<b>backoff</b>	intentionally increased delay or reduced rate (typically of retransmissions)
<b>bent pipe relay</b>	analog signal repeater (typically a satellite): no FEC or traffic switching
<b>BER</b>	Bit Error Rate
<b>bps</b>	Bits Per Second
<b>burst error</b>	error sequence due to auto-correlated noise or interfering transmissions
<b>C3</b>	Command, Control & Communications
<b>C4ISR</b>	C3 + Computers + ISR
<b>CAP</b>	Channel Access Protocol
<b>CERBS</b>	Compressed EHF Report-Back System
<b>CMR</b>	Concurrent Multipath Routing
<b>COP</b>	Common Operational Picture
<b>COTS</b>	Commercial Off The Shelf
<b>CRIU</b>	CAP Router Interface Unit
<b>currency</b>	relative timeliness of information or message receipt
<b>CWSP</b>	Commercial Wideband SATCOM Program
<b>DAMA</b>	Demand Assigned Multiple Access
<b>DASA</b>	Demand Assigned Single Access
<b>data corruption</b>	unintentional modification of data (typically due to errored or lost transmissions)
<b>Delta Load</b>	a standard suite of COTS/GOTS software used aboard submarines
<b>disadvantaged</b>	severely performance limited or degraded
<b>DL-ARQ</b>	Data Link (OSI Layer 2) ARQ
<b>DMR</b>	Digital Modular Radio
<b>dropout error</b>	sequence of errors due to temporary loss of signal strength
<b>DSCS</b>	Defense Satellite Communication System
<b>dump</b>	a file of accumulated data (typically a log of monitoring data)
<b>dupe ACK</b>	duplicate ACK of the same transmission as a previous ACK
<b>DiffServ</b>	Differentiated Services (RFC 2475)
<b>EAM</b>	Emergency Action Message
<b>EHF</b>	Extremely High Frequency
<b>ELF</b>	Extremely Low Frequency
<b>ELN</b>	Explicit Loss Notification
<b>EMCON</b>	EMissions CONTROL (typically radio silence)
<b>end points</b>	ultimate sending and receiving points of a communication
<b>FDMA</b>	Frequency Division Multiple Access: an approach to MAC
<b>FEC</b>	Forward Error Correction
<b>FTP</b>	File Transfer Protocol
<b>full duplex</b>	capability for traffic to flow simultaneously in both directions

<b>gateways</b>	intermediate systems that enable communications between endpoints
<b>GFCP</b>	Generic Front-end Communication Processor
<b>GOTS</b>	Government Off The Shelf
<b>GUI</b>	Graphical User Interface
<b>half duplex</b>	capability for traffic to flow in either direction, but not in both at once
<b>HDR</b>	High Data Rate
<b>HF</b>	High Frequency
<b>HMI</b>	Human Machine Interface
<b>IETF</b>	Internet Engineering Task Force: develops Internet standards etc.
<b>integrity</b>	absence of error or corruption
<b>IP</b>	Internet Protocol
<b>IRTT</b>	Initial Round Trip Time
<b>ISABPS</b>	Integrated Submarine Automated Broadcast Processing System
<b>ISO</b>	International Standards Organization
<b>ISR</b>	Intelligence, Surveillance & Reconnaissance
<b>LAN</b>	Local Area Network
<b>latency</b>	time delay between transmission and reception of a communication
<b>layer</b>	OSI model protocol design partition, grouping similar or related functions
<b>Layer 1</b>	Physical: hardware (modems, radios, etc.) and transmission media
<b>Layer 2</b>	Data Link: single hop communication between adjacent network modes
<b>Layer 3</b>	Network: concatenates Data Links to enable Transport
<b>Layer 4</b>	Transport: end-to-end multi-hop communications
<b>Layer 5</b>	Session: domain name lookup, user authentication, etc.
<b>Layer 6</b>	Presentation: format and structure of application data
<b>Layer 7</b>	Application: software communication end-point
<b>LDR</b>	Low Data Rate
<b>LF</b>	Low Frequency
<b>LPD</b>	Low Probability of Detection
<b>LPI</b>	Low Probability of Intercept
<b>MAC</b>	Medium Access Control: enables sharing of a medium by multiple stations
<b>Mbps</b>	MegaBits Per Second
<b>MDR</b>	Medium Data Rate
<b>MDT</b>	Message Distribution Terminal
<b>MF</b>	Medium Frequency
<b>NACK</b>	Negative ACK: indicates a transmission was <i>not</i> received successfully
<b>NB</b>	Narrow Band
<b>NBSV</b>	NB Secure Voice
<b>NIPRNET</b>	Non-secure IP Router NETwork
<b>offered load</b>	traffic that endpoints are trying to move through a network
<b>OODD</b>	Out Of Order Delivery
<b>OSI</b>	Open Systems Interconnection project of the ISO



<b>packet</b>	generic term for a message, message segment, transmission block, etc.
<b>PC</b>	Personal Computer
<b>PDU</b>	Protocol Data Unit: packet exchanged between local and remote peers
<b>PPP</b>	Point-to-Point Protocol
<b>PTP</b>	Point-To-Point
<b>QoS</b>	Quality Of Service
<b>quality critical load</b>	offered load with a need for assurance or other QoS (versus low latency)
<b>reliable delivery</b>	confidence (vs confirmation) that transmissions are received successfully
<b>RF</b>	Radio Frequency
<b>RFC</b>	Request For Comment: standard, etc. publication of the IETF
<b>RFDAC</b>	RF Distribution And Control
<b>RFIU</b>	RF Interface Unit
<b>RUDP</b>	Reliable UDP
<b>RST</b>	Reset - Tear down the connection
<b>SACK</b>	Selective ACK
<b>SatCom</b>	SATellite COMmunication
<b>SCPS</b>	Space Communications Protocol Standards
<b>SCPS-TP</b>	SCPS Transport Protocol
<b>SCPS-TP GW</b>	SCPS-TP gateway: transparently translates TCP <-> SCPS-TP
<b>SDU</b>	Service Data Unit: packet exchanged between adjacent local stack layers
<b>silly state</b>	any state not intentionally visited but possibly reachable (typically by error)
<b>simplex</b>	capability for traffic to flow always in one direction only
<b>SMART</b>	Submarine Message Automated Routing Terminal
<b>SMB</b>	Submarine Message Buffer
<b>SMS</b>	Single Messaging System
<b>SNACK</b>	Selective NACK
<b>SRB</b>	Submarine Report-Back
<b>SubHDR</b>	Submarine HDR
<b>TCP</b>	Transmission Control Protocol: provides assured data stream delivery
<b>TDMA</b>	Time Division Multiple Access: an approach to MAC
<b>time critical load</b>	offered load with a need for low latency (versus high quality)
<b>UDP</b>	User Datagram Protocol: provides non-assured packet delivery
<b>UHF</b>	Ultra High Frequency
<b>VHF</b>	Very High Frequency
<b>VLF</b>	Very Low Frequency
<b>WAN</b>	Wide Area Network
<b>WB</b>	Wide Band

## **Appendix B: Proxy Caching Verification Tests**

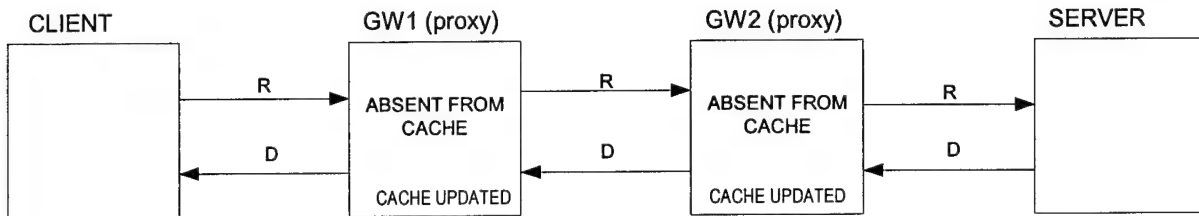
This appendix presents results of tests performed to ascertain the actions of the Apache Server in response to Web page requests under differing states of the caches of the onboard (GW1) and shore (GW2) proxy servers. The tests were performed in two general states; offline, in which there is no connection beyond the boat (i.e., to GW2); and online, in which the boat is connected via either cable or wireless means to GW2 on the shore. There are three informational states for both gateway caches: requested page absent, stale or current; and two temporal states: timer expired and timer not expired. The results for the condition in which there is no connection to the shore are easy to document: if the page is absent from the GW1 cache, an error condition of "file not found" is returned; if the page is present, it is retrieved as is (currency of the page cannot be verified). The remainder of this appendix illustrates the reaction under various conditions when connected to the shore (GW2).

In the charts, a "D" (data) return results in a (stale) file's being replaced in the cache with the current file or a new file's being added to the cache, along with an update of "date/time last accessed". An "S" (status: not modified) return indicates that the file in the cache is current and results in only the "date/time last accessed" information being updated in the cache for future currency checks.

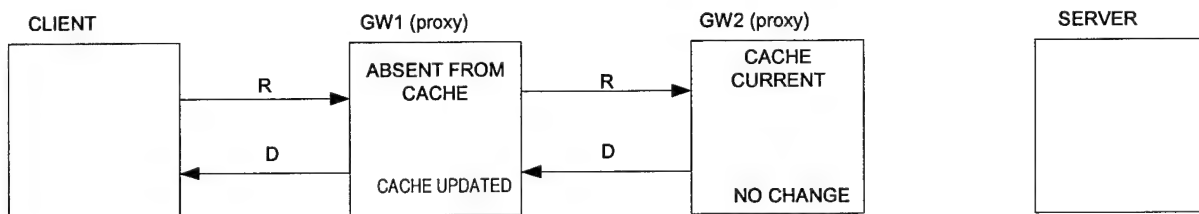
For these tests, the client was configured to not cache.

**Test Results for scenarios 1-9 - the cache timer has not expired on either cache:**

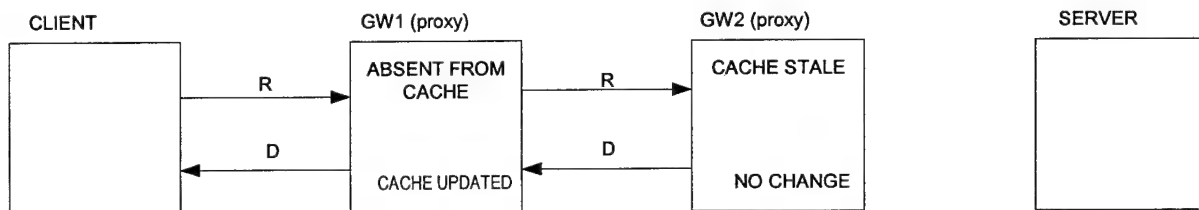
**Scenario 1**



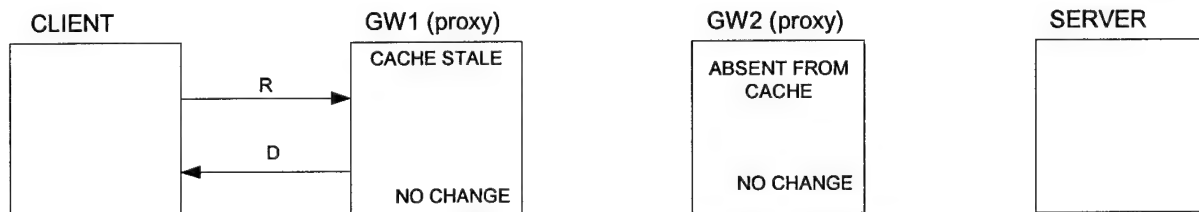
**Scenario 2**



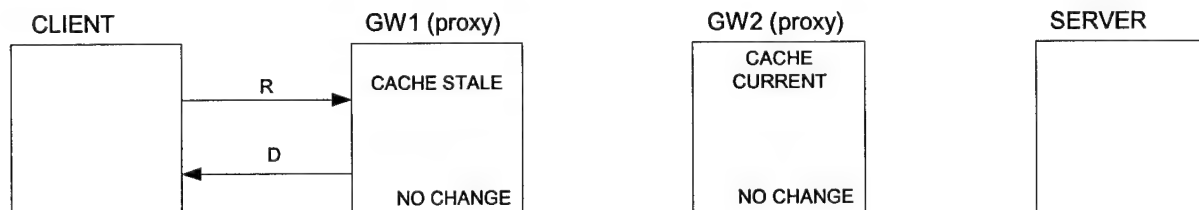
**Scenario 3**



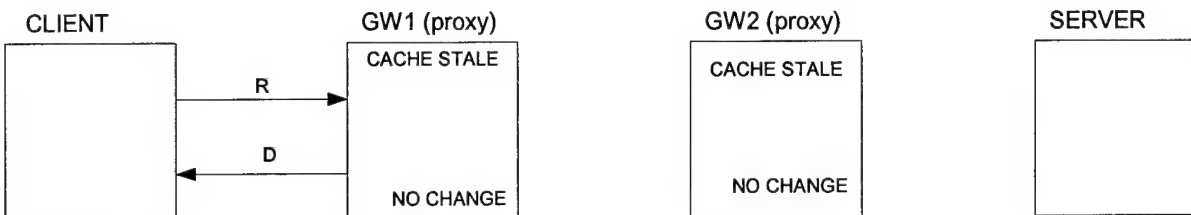
**Scenario 4**



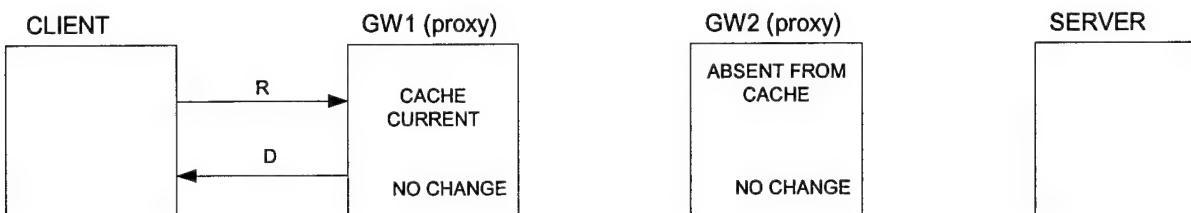
**Scenario 5**



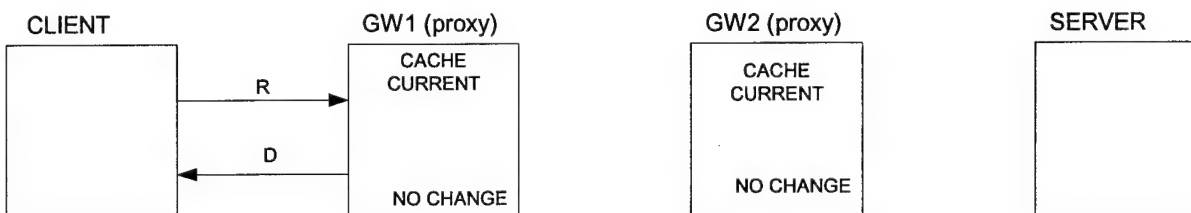
Scenario 6



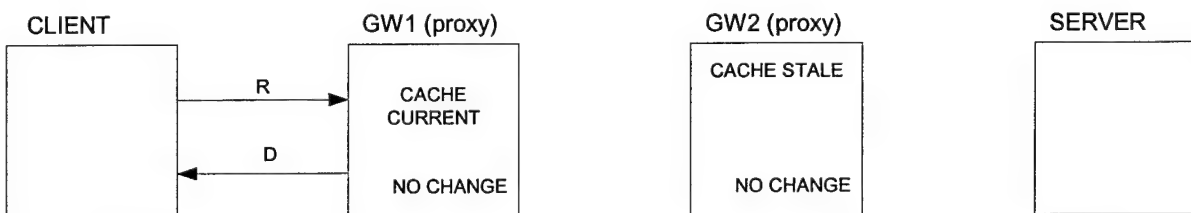
Scenario 7



Scenario 8

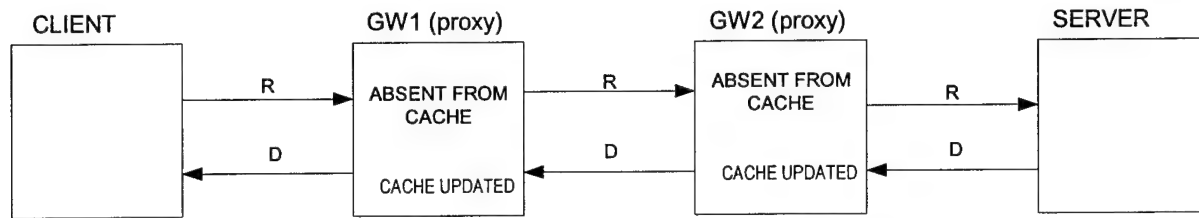


Scenario 9

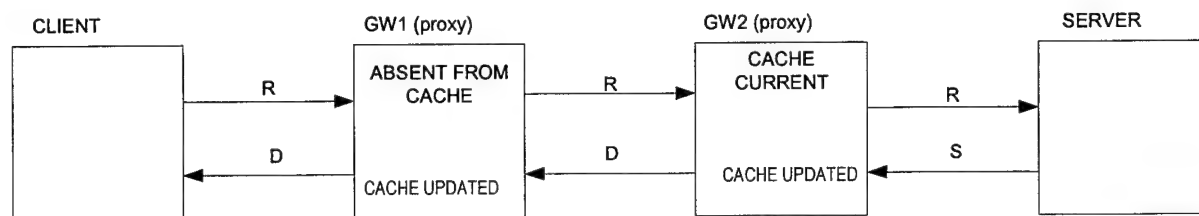


**Test results for Scenarios 10-18 – the timer on both caches has expired:**

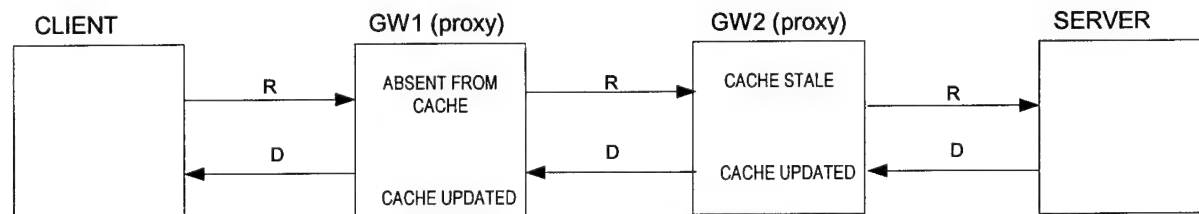
**Scenario 10**



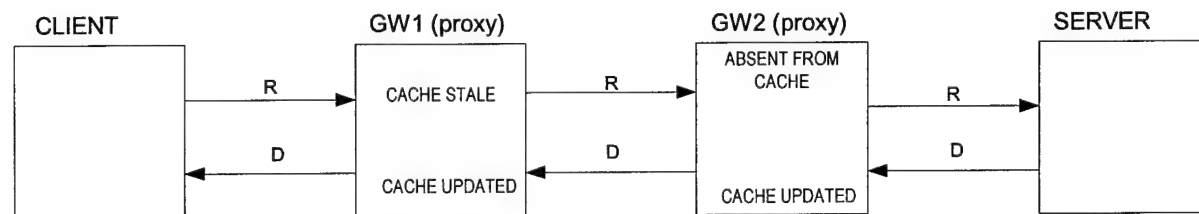
**Scenario 11**



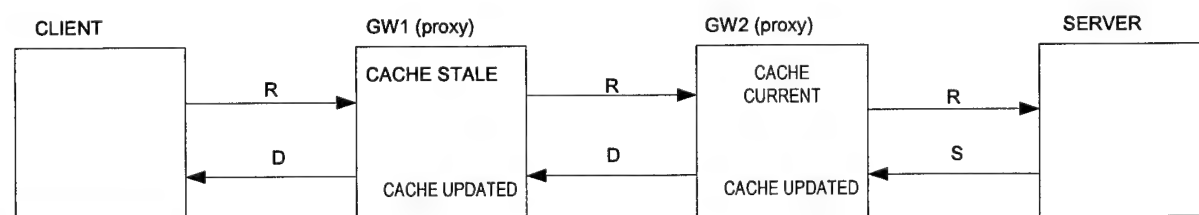
**Scenario 12**



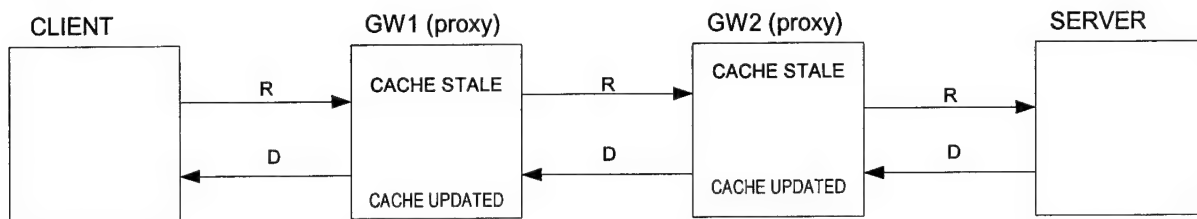
**Scenario 13**



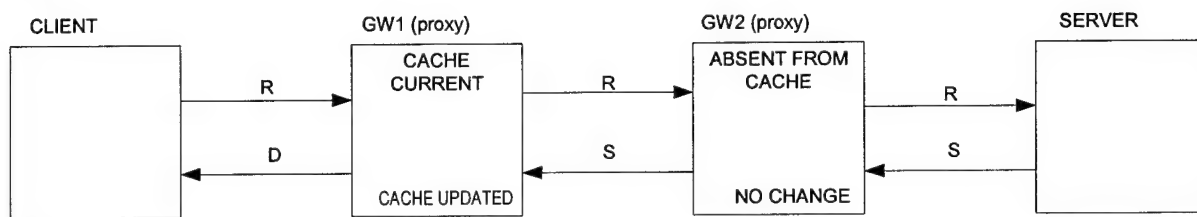
**Scenario 14**



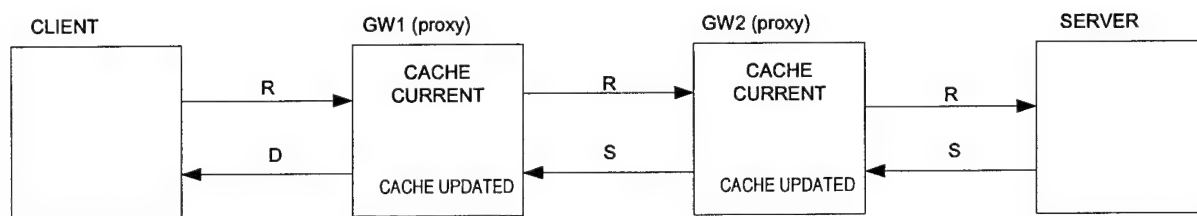
Scenario 15



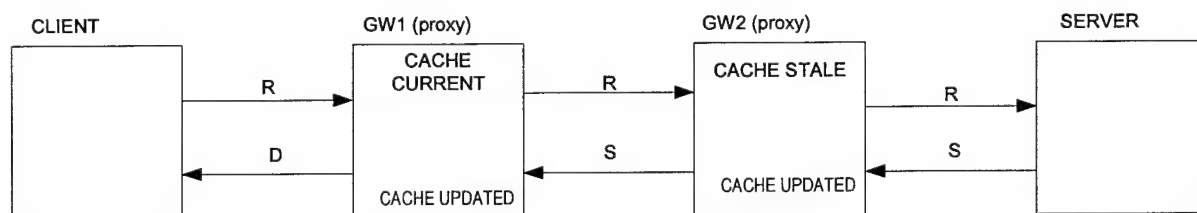
Scenario 16



Scenario 17



Scenario 18





## Appendix C: APACHE WEB SERVER with proxy+cache+offline HOWTO

The following appendix provides documentation of the procedure followed in creating the offline/online caching environment in the Critical Technologies lab. Apache Web Server was installed and tested using both online and offline capabilities. Complete test results are documented in Appendix A.

### (1) Downloading the sources

- Download the apache server from <http://www.apache.org> ( version > 1.3.x )  
in directory "/usr/src".  
i.e. /usr/src/apache\_1.3.19.tar.gz
- Download the patch file for 'offline' mode from  
<http://dSPACE.dial.pipex.com/david.whitmarsh/offline.html> (offline-1.3.14.patch)  
in directory "/usr/src".  
i.e. /usr/src/offline\_1.3.14.patch

### (2) Untar the source codes

- Untar the apache server source codes downloaded.  
%> cd /usr/src/  
%> tar -zxvf apache\_1.3.19.tar.gz  
It will create "/usr/src/apache\_1.3.19/" directory
- Apply patch file for offline mode into apache server source codes.  
%> cd /usr/src/apache\_1.3.19/src/modules/proxy/  
%> patch -p1 < /usr/src/offline\_1.3.14.patch

### (3) Compiling the source codes

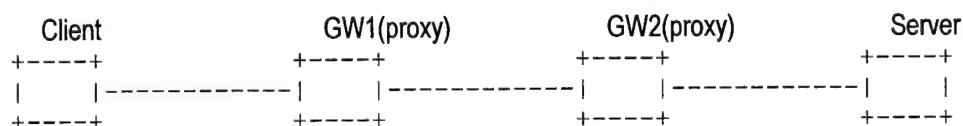
```
%> cd /usr/src/apache_1.3.19/
%> ./configure -v --enable-rule=SHARED_CORE --enable-module=most --enable-shared=max
%> make
```

### (4) Installing the apache web server

```
%> cd /usr/src/apache_1.3.19/
%> make install
It will install the server in "/usr/local/apache/" directory.
```

### (5) Configuring the apache server

Changes should be applied on the proxy boxes based on the network diagram below,



- Changed the owner and group of proxy cache directory

```
%> chown apache /usr/local/apache/proxy
chgrp apache /usr/local/apache/proxy
```

- Edit and change "/usr/local/apache/conf/httpd.conf" as follows.

```
...
Port 8080
User apache
Group apache
```

```
...
ServerAdmin root@localhost
ServerName localhost
```

```
...
CacheNegotiatedDocs
```

```
...
<IfModule mod_proxy.c>
ProxyRequests On
```

```
/* This line is needed only at GW1 */
ProxyRemote * http://GW2:8080
```

```
    <Directory proxy:>
        Order deny,allow
        Allow from all
    </Directory>
    ProxyVia On
    CacheRoot "/usr/local/apache/proxy"
    CacheSize 50000
    CacheGcInterval 4
    CacheMaxExpire 24
    CacheLastModifiedFactor 0.1
    CacheDefaultExpire 1
    #NoCache a_domain.com another_domain.edu joes.garage_sale.com
    CacheMinGcAge 168
    KeepCache Server
    ProxyCacheForOffline On
```

```
/* If offline mode: On, If not: Off */
ProxyOffline Off
</IfModule>
```

```
...
```

**(6) Start the httpd server**

- Make the following "/usr/local/apache/bin/httpd.init".
- Make "/usr/local/apache/bin/httpd.init" executable.  
%> chmod 777 /usr/local/apache/bin/httpd.init
- Start the httpd server  
%> /usr/local/apache/bin/httpd.init start

```
----- httpd.init -----

#!/bin/bash
#
# Startup script for the Apache Web Server
#
# chkconfig: - 85 15
# description: Apache is a World Wide Web server. It is used to serve \
#              HTML files and CGI.
# processname: httpd
# pidfile: /usr/local/apache/logs/httpd.pid
# config: /usr/local/apache/conf/access.conf
# config: /usr/local/apache/conf/httpd.conf
# config: /usr/local/apache/conf/srm.conf

# Source function library.
. /etc/rc.d/init.d/functions

# This will prevent initlog from swallowing up a pass-phrase prompt.
INITLOG_ARGS=""

# Source additional OPTIONS if we have them.
if [ -f /etc/sysconfig/apache ] ; then
    . /etc/sysconfig/apache
fi

# Path to the httpd binary.
httpd=/usr/local/apache/bin/httpd
prog=httpd
RETVAL=0

# Change the major functions into functions.
moduleargs() {
    moduledir=/usr/local/apache/libexec
    moduleargs=
    /usr/bin/find ${moduledir} -type f -perm -0100 -name "*.so" | awk '{\
        gsub(".*", "");\
        gsub("^mod_", "");\
        gsub("^lib", "");\
        gsub("\.so", "");\
        print "-DHAVE_" toupper($0)}'
    echo ${moduleargs}
}
```

```

}
start() {
    echo -n $"Starting $prog: "
    daemon $httpd `moduleargs` $OPTIONS
    RETVAL=$?
    echo
    [ $RETVAL = 0 ] && touch /usr/local/apache/logs/httpd.lock
    return $RETVAL
}
stop() {
    echo -n $"Stopping $prog: "
    killproc $httpd
    RETVAL=$?
    echo
    [ $RETVAL = 0 ] && rm -f /usr/local/apache/logs/httpd.lock /usr/local/apache/logs/httpd.pid
}

# See how we were called.
case "$1" in
    start)
        start
        ;;
    stop)
        stop
        ;;
    status)
        status $httpd
        ;;
    restart)
        stop
        start
        ;;
    reload)
        echo -n $"Reloading $prog: "
        killproc $httpd -HUP
        RETVAL=$?
        echo
        ;;
    condrestart)
        if [ -f /usr/local/apache/logs/httpd.pid ] ; then
            stop
            start
        fi
        ;;
    *)
        echo $"Usage: $prog {start|stop|restart|reload|condrestart|status}"
        exit 1
esac

exit $RETVAL

```

## **(7) Configure the Proxy Server**

### **(7.1) In case of Explicit Proxy Server**

Set up the browser in Client to use proxy server as GW1:8080.

### **(7.2) In case of Transparent Proxy Server**

#### **(7.2.1) Set up the transparent proxy server in GW1**

- download the source code from `ftp://ftp.nlc.net.au/pub/unix/transproxy/`  
in `/usr/src/` directory  
i.e. `/usr/src/transproxy-1.4.tgz`
- untar the source code  
`%> cd /usr/src`  
`%> tar -zxvf transproxy-1.4.tgz`
- compile the source code  
`%> cd /usr/src/transproxy-1.4`  
`%> make`
- install the transparent proxy server  
`%> cd /usr/src/transproxy-1.4`  
`%> make install`  
it will install the server in `/usr/local/sbin /usr/local/man` directory.
- Run the transparent proxy server  
`%> /usr/local/sbin/tproxy -s 80 -r nobody localhost 8080`  
  
it tells that the transparent proxy server to accept requests on port 80  
and to pass these on to the host apache proxy at port 8080

#### **(7.2.2) Set up the netfilter rules in GW1 to forward incoming data to the transparent proxy server (port 80).**

- if you use ipchains  
`ipchains -A input -i eth0 -p tcp -s 0.0.0.0/0 -d 0.0.0.0/0 80 -j REDIRECT 80`
- if you use iptables  
`iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 80 -j REDIRECT --to-port 80`

## **Appendix D: Linux IP Filtering – Comparison of IP Chains and IP Tables**

### **Introduction**

This document will present a brief overview of the capabilities available between IP Chains (Linux 2.2) and IP Tables (Linux 2.4). As a result of this research, IP Tables was chosen as the packet filtering mechanism of choice for use in the proposed system.

### **Description**

Although traditionally a packet filter is used to allow or drop a packet, the fields that the filter inspects can also be used to produce a firewall mark used by the kernel's routing daemon for prioritization of network traffic.

### **IP Chains**

IP Chains operates on packets by comparing the packet header to a sequence of rules known as a chain. If a match is found, it decides what to do based on the policy defined by the rule. In the event, a packet does not match any rules, it will follow the default policy for that chain.

### **Targets**

IP Chains supports six targets (policies):

- ACCEPT – allow the packet through
- REJECT – drop the packet with ICMP message to sender
- DENY – drop the packet without notification
- MASQ – masquerade the packet
- REDIRECT – send the packet to a local port instead of its specified destination
- RETURN – identical to falling off of the end of chain
- User-defined chain – jump to a user defined chain

### **Filtering Criterion**

- Source and destination IP Addresses
- Protocol
- Source and destination ports (TCP/UDP)
- ICMP types and codes
- Interface
- TCP SYN only
- Type of Service bits

### **IP Tables**

IP Tables is based on the same chain-based operation as IP Chains. It is designed to be more flexible and extensible than IP chains, introducing new criteria for rule matching.

### **Targets**

- DROP – packet is dropped with no feedback to sender
- ACCEPT – packet is allowed
- LOG – logging of matched packets
- REJECT – packet is dropped but ICMP message is returned to sender
- RETURN – same effect as falling off of the end of a chain, utilizing the default policy
- QUEUE – queues the packet for handling by a user-space application
- User-defined chain – jump to a user defined chain

### **Filtering Criteria**

- Source and destination IP Addresses
- Protocol
- Source and destination ports (TCP/UDP)
- ICMP types and codes
- Interface
- TCP flags
- Type of Service bits
- MAC Address
- Rate limits
- Owner of packet creator
- Unclean (experimental sanity checking)
- State match – connection tracking analysis

### **Conclusion**

Due to IP Tables extended criteria for rule matching, we recommend its use as the preferred packet filtering mechanism. However, IP Chains contains most of IP Tables functionality and would be an excellent fallback if implementation details require a 2.2 kernel.